IDS

薛静锋　宁宇鹏　阎　慧　编著

入侵检测技术

国家信息化安全教育认证(ISEC)系列教材

机械工业出版社
CHINA MACHINE PRESS

ISBN 7-111-14166-0/TP · 3515

◎封面设计 旭洲企划 刘吉维

国家信息化安全教育认证（ISEC）系列教材

- 网络安全基础
- 防火墙原理与技术
- 入侵检测技术
- PKI 技术
- VPN 技术
- 数据备份与灾难恢复
- 网络隔离与网闸
- 信息安全法规与标准
- 信息安全策略与机制
- 信息安全团队构建与管理

ISBN 7-111-14166-0
9 787111 141662 >

定价：20.00元

地址：北京市百万庄大街22号 邮政编码：100037
联系电话：(010) 68326294 网址：http://www.cmpbook.com
E-mail:online@cmpbook.com

国家信息化安全教育认证(ISEC)系列教材

入侵检测技术

薛静锋　宁宇鹏　阎　慧　编著

机械工业出版社

本书作为国家信息化安全教育认证(ISEC)系列教材中的一本，全面介绍了入侵检测技术，包括入侵检测基础知识，入侵检测系统，入侵检测技术，入侵检测系统的性能指标和评估标准，主要入侵检测系统分析及入侵检测的标准化工作，入侵检测系统的实现，Snort 分析以及入侵检测技术的发展趋势。

本书不仅适合大专院校相关专业作为教材，对从事信息和网络安全方面的管理人员和技术人员也有参考价值。

图书在版编目(CIP)数据

入侵检测技术/薛静锋等编著. —北京：机械工业出版社，2004.4
(国家信息化安全教育认证(ISEC)系列教材)
ISBN 7-111-14166-0

Ⅰ.入… Ⅱ.薛… Ⅲ.计算机网络—安全技术—资格考核—教材 Ⅳ.TP393.08

中国版本图书馆 CIP 数据核字(2004)第 019303 号

机械工业出版社(北京市百万庄大街 22 号 邮政编码 100037)
责任编辑：李馨馨
责任印制：施 红

北京铭成印刷有限公司印刷·新华书店北京发行所发行

2004 年 4 月第 1 版·第 1 次印刷
787mm × 1092mm $^{1}/_{16}$·11 印张·270 千字
0 001—5 000 册
定价：20.00 元

国家信息化安全教育认证(ISEC)专家组

卿斯汉　中国科学院信息安全技术工程研究中心主任　研究员
曲成义　中国航天科技集团公司第710研究所总工　研究员
许榕生　中国科学院高能物理研究所计算中心研究员
贾颖禾　国务院信息化工作办公室网络与信息安全组研究员
曹元大　北京理工大学软件学院院长　博士生导师
杨义先　北京邮电大学信息安全中心主任　博士生导师
林　鹏　国家计算机网络应急技术处理协调中心广东分中心副主任
　　教授级高级工程师
祁　金　公安部公共网络信息安全监察局管理监察处副处长
井乾元　公安部公共网络信息安全监察局安全对策处副处长
万平国　国际信息战略研究中心理事　中网通讯网络有限公司董事长
刘宝旭　中国科学院高能物理研究所计算中心副研究员

教材编委会

主　任: 宋　玲
副主任: 赵小凡　张会生　欧阳满　蔡金荣　沈志工
成　员: 洪京一　张宝泰　王　宏　孙论强　彭　澎　张晓伟
　　刘树安　刘　旸　马志谦　胡　铮　宁宇鹏　阎　慧
　　王　伟　薛静锋　辛　阳

出版说明

随着信息化在我国的不断深入和发展，信息技术和网络给社会的经济、科教、文化和管理等各个方面注入了新的活力。人们在感受它所带来的新体验、享受它为我们带来高效率的同时，也面临着日益突出的信息安全问题。党和国家领导人多次强调，必须充分认识到做好信息安全保障工作的重要性，大力搞好技术开发和人才培养。

对信息安全专业人才的需求是多层次的。从信息安全基础理论研究到新技术的开发利用，再到各级网络信息系统信息安全保障体系的建设、运行，需要根据不同要求有针对性的进行人才培养。

国家信息化安全教育认证(ISEC)项目是由信息产业部信息化推进司推出的信息安全领域的国家级认证体系。该项目由中国电子商务协会监督，由 ISEC 国家信息化安全教育认证管理中心统一管理与实施。ISEC 国家信息化安全教育认证管理中心以行业为基础、以技术为核心制定了一套面向应用的教育方案。根据工作性质的不同，教育对象被划分为规划决策层、管理运营层和操作层三个层次。认证体系的设计，课程内容及相应的教学考试大纲的编写和指定是针对三个层次人员对信息安全知识和技能的不同需求和理解程度的不同而制定的。确保不同层次，不同需求的各类人员从各自的角度充分掌握和理解信息安全的知识和技能。

本系列教材是在国家信息化安全教育认证(ISEC)专家组的指导下，由国家信息化安全教育认证(ISEC)教材编委会组织编写的。始终以 ISEC 国家信息化安全教育认证管理中心制订的各级考试大纲为依据，坚持面向行业用户的需求和侧重技术应用两个基本原则，全面地介绍了信息安全各种主流技术和管理规范，以帮助读者深入了解信息安全本质，并熟练掌握相应的技能，从而建立完备的信息安全观念。本系列教材包括：《网络安全基础》、《防火墙原理与技术》、《入侵检测技术》、《VPN 技术》、《PKI 技术》、《数据备份与灾难恢复》、《网络隔离与网闸》、《信息安全法规与标准》、《信息安全策略与机制》、《信息安全团队构建与管理》，共计 10 本。

在写作过程中，北京正阳天马信息技术有限公司为本系列教材的编写提供了很多宝贵的建议和支持。

前　言

如今，网络安全问题越来越受到人们的关注，也逐渐成为各相关科研机构研究的热点。传统的网络安全技术以防护为主，即采用以防火墙为主体的安全防护措施。但是，面对网络大规模化和入侵复杂化的发展趋势，以防火墙技术为主的被动防御技术越来越力不从心，由此产生了以入侵检测技术为主的主动保护技术。

入侵检测技术是网络安全的核心技术之一，它通过从计算机网络或计算机系统中的若干关键点收集信息并对其进行分析，从而发现网络或系统中是否有违反安全策略的行为和遭到袭击的迹象。利用入侵检测技术，不但能够检测到外部攻击，而且能够检测到内部攻击或误操作。但是入侵检测技术毕竟还是一门新技术，还处在不断发展的过程中。本书以国家信息化安全教育认证(ISEC)考试大纲为依据，重点讲解了入侵检测的有关理论知识、技术原理和应用案例。全书包括 8 章内容，第 1 章主要介绍入侵检测的相关基础知识。包括入侵检测的产生与发展历程、入侵检测的基本概念、作用以及研究入侵检测的必要性。第 2 章主要介绍关于入侵检测系统的相关知识。包括入侵检测系统的基本模型、入侵检测系统的工作模式和分类方法、入侵检测系统的数据源以及入侵检测系统的部署方式。第 3 章主要介绍入侵检测技术。包括入侵检测的过程、入侵分析的概念和入侵分析的模型、入侵分析的方法，并且分析了入侵检测中的各种告警与响应方式，此外还对入侵追踪进行了比较详细的介绍。第 4 章介绍入侵检测系统的性能指标和评估标准。包括影响入侵检测性能的参数、评价检测算法性能的测度和评价入侵检测系统性能的标准，在此基础上，介绍了关于网络入侵检测系统的测试评估、测试环境和测试软件，另外还对入侵检测评估现状进行了分析。第 5 章介绍主要的入侵检测系统以及入侵检测的标准化工作。首先对国外主要入侵检测系统进行了介绍和分析；接着对入侵检测的标准化工作进行了详细介绍，主要是 CIDF 的标准化工作和 IDWG 的标准化工作。第 6 章以一个基于 Agent 的分布式入侵检测系统的设计与实现为例，介绍入侵检测系统的实现过程。第 7 章对开放源代码的入侵检测软件 Snort 进行了详细的分析。第 8 章对入侵检测的发展趋势进行了简要分析。分析了入侵检测技术的现状，目前的技术趋势，未来的安全趋势，以及入侵检测的前景。

本书由北京理工大学薛静锋、宁宇鹏、阎慧执笔完成，其中第 1、5、6 章由薛静锋编写，第 4、7、8 章由宁宇鹏编写，第 2、3 章由阎慧编写。本书在写作过程中得到了王勇博士、王伟博士、李志强老师以及北京正阳天马信息技术有限公司刘旸先生、马志谦先生的热情帮助，在此一并表示感谢。

编　　者

2004 年 3 月

目　录

第1章　入侵检测基础知识

本章导读：

本章主要介绍入侵检测的基础知识。首先介绍了入侵检测的产生与发展，包括入侵检测的早期研究、基于主机与基于网络的入侵检测的研究以及两者的集成；接着介绍了入侵检测的基本概念，包括入侵检测的概念、作用以及研究入侵检测的必要性。

“美国八大著名网站被黑、克林顿总统亲自召集网络安全会议并拨款20亿美元”这一消息，使各国政府、IT厂商和业界同仁在感到震惊的同时，也开始思考网络安全问题，并采取了必要的行动。其实，网络安全的警钟早已鸣响，而且还要警钟长鸣。

根据美国FBI的调查，美国每年因为网络安全造成的经济损失超过170亿美元。75%的公司报告财政损失是由于计算机系统的安全问题造成的。但只有17%的公司愿意报告黑客入侵，大部分公司由于担心负面影响而不愿声张。在所有的损失中虽然只有59%可以定量估算，但每个组织的平均损失已达40万美元之多。

对于企业网络来说，入侵的来源可能是企业内部心怀不满的员工、网络入侵者，甚至是竞争对手。攻击者可以窃听网络上的信息，窃取用户的口令、数据库的信息，还可以篡改数据库内容，伪造用户身份，否认自己的签名。更有甚者，攻击者可以删除数据库的内容，摧毁网络节点，释放计算机病毒，致使整个企业网络陷入瘫痪。

那么网络在被动保护自己不受侵犯的同时，能否采取某些技术，主动保护自身的安全呢？入侵检测技术就是主动保护自己免受攻击的一种网络安全技术，而入侵检测系统（Intrusion Detection System，IDS）就是能够实施入侵检测的系统。入侵检测技术是网络安全体系的一种防范措施。

1.1　入侵检测的产生与发展

20世纪70年代，随着计算机的速度、数量的增长以及体积的减小，对计算机安全的要求显著增加。面对这样的形势，在1977年和1978年，美国国家标准局召开了有政府和商业组织代表参加的会议，就当时的安全、审计和控制状况提出报告。

与此同时，军用系统中计算机的使用范围迅速扩大，出于对安全问题的考虑，美国国防部提高了计算机审计的详细程度并以此作为一项安全机制。这个项目由James Anderson负责。

1.1.1　早期研究

1980年，James Anderson在写给一个保密客户的技术报告中指出，审计记录可以用于识别计算机误用。他提出了入侵尝试（intrusion attempt）或威胁（threat）的概念，并将其定义为：潜在、有预谋且未经授权而访问信息、操作信息，致使系统不可靠或无法使用的企图。同时，他给威胁进行了分类，并对审计子系统提出了改进意见，以使该系统可以检测误用。他认为审计记录分析可

以监视入侵行为,并对入侵进行分类,还提出对不同用户的不同渗透方法。如表 1-1 所示。

表 1-1 不同用户的不同渗透方法

	授 权	非授权
外部用户		外部渗透
内部用户	不当行为	内部渗透

James Anderson 主要工作对象是重要的分级客户,这些客户在主机环境中处理敏感数据,其特点是有严格的安全管理控制。客户有审计所有计算机活动的需求,并由安全部门职员手工检查审计跟踪并调查在审计跟踪中未发现的问题以支持该策略。随着计算量的增加,手工检查和调查工作变得繁重不堪。

James Anderson 在一段时间内致力于解决"伪装者"的问题,"伪装者"指那些用盗窃来的用户名和密码访问系统的人。对系统而言,"伪装者"似乎是合法用户。James Anderson 建议一些针对某些用户行为的统计分析应当具备识别系统不正常使用模式的能力,这是发现"伪装者"的一种方法。

1983 年,SRI(Stanford Research Institute)用统计方法分析 IBM 大型机的 SMF(System Management Facility)记录,这也是早期对入侵检测的研究。

由于 20 世纪 80 年代初期网络还没有像今天这样普遍和复杂,网络之间也没有完全互联,因此关于入侵检测的研究主要是基于主机的事件日志分析。而且由于入侵行为在当时是相当少见的,因此入侵检测在早期并没有受到人们的重视。

1.1.2 主机 IDS 研究

1986 年,SRI 的 Dorothy E.Denning 发表了一篇论文《An Intrusion-Detection Model》,该文深入探讨了入侵检测技术,探索了行为分析的基本机制,首次将入侵检测的概念作为一种计算机系统安全防御措施而提出,并且建立了一个独立于系统、程序应用环境和系统脆弱性的通用入侵检测系统模型,如图 1-1 所示。这篇文章被认为是 IDS 的开山之作,与传统的加密和访问控制相比,IDS 是全新的计算机安全措施。

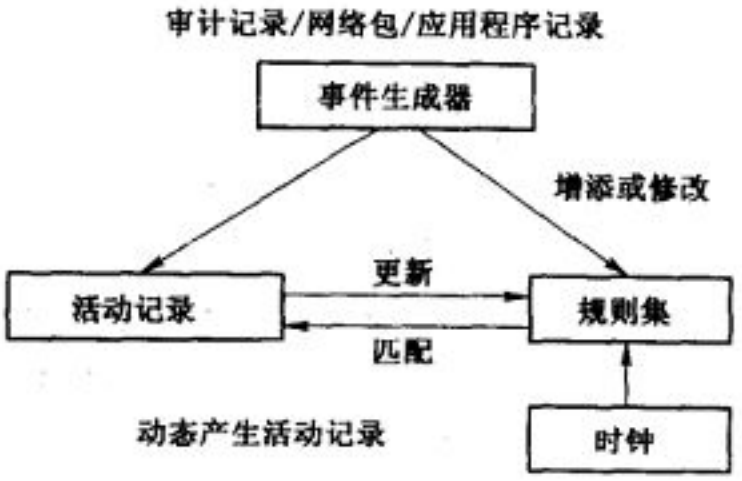

图 1-1 通用入侵检测模型

1988 年,SRI 开始开发 IDES(Intrusion Detection Expert System)原型系统,它是一个实时入侵检测系统。它采用了统计技术来进行异常检测,用专家系统的规则进行误用检测。IDES 在实现双重分析(分析和异常检测)和实时分析两个方面迈出了关键的一步。该系统被认为是

入侵检测研究中最有影响的一个系统，也是第一个在一个应用中运用了统计和基于规则两种技术的系统。

从 1992 年到 1995 年，SRI 对 IDES 在原有基础上加强了优化，在以太网的环境下实现了产品化的 NIDES(Next-Generation Intrusion Detection Expert System)，它具有 IDES 的双重分析特性，采用更为通用、灵活的方法，对于目标系统和审计数据的类型没有限制，采用 C/S 模式。但是在规模化和针对网络环境使用方面还有所欠缺，而且缺少协同工作的能力。由于把用户作为分析的目标(或者说单元)，因此对于多域联合攻击无能为力。

1988 年，针对美国空军计算机系统的多用户环境，Los Alamos 国家实验室的 Tracor Applied Sciences 和 Haystack Laboratories 采用异常检测和基于 Signature 的检测，开发了 Haystack 系统，该系统主要用于检测 Unisys 大型主机。与以往的系统不同，该系统建立了两个模型：为每个用户建立的用户模型和通用用户模型。

同时，出现了为美国国家计算机安全中心 Multics 主机开发的多入侵检测及告警系统(Multics Intrusion Detection and Alerting System，MIDAS)，该系统是在国家计算机安全中心的公共信息系统 Dockmaster 上应用的第一个入侵检测系统。

1989 年，Los Alamos 国家实验室的 Hank Vaccaro 为国家计算机安全中心(National Computer Security Center，NCSC)和能源部(Department of Energy，DOE)开发了 W&S (Widsom and Sence)系统，这是一个基于主机的异常检测系统。W&S 处理一个训练用的数据集，并产生用于描述数据特征的元规则(metarule)，随后当系统应用于新的数据集时，该系统就用这些规则检测异常。W&S 最初被设计用于检测存储核材料的数据记录中的异常，后来被修改为检测 VMS 操作系统的审计记录，以及检测人为的误操作。由于检测到的异常和人为的误操作混合在一起，所以 W&S 永远都不能应用于生产环境中，因为构造、修剪元规则树的方式使得很难解释得到的结果。

同年，PRC(Planning Research Corporation)公司开发了 ISOA(Information Security Officers Assistant)，它由一套统计工具、一个专家系统和一套分级的“利害关系级别(concern levels)”组成。其技术基于一种称为迹象与警告(Indications and Warnings，I&W)的模型，应用该模型可以对即将发生的攻击预先告警。引入的审计数据与一组期望的迹象相比较，并按层次排列以反映利害关系级别。异常是用三类参数来检测的：用户、节点及整个系统。ISOA 后来用在了 PRC 入侵检测系统 PreCis 中。

以上研究虽然有的是在局域网环境下展开的，但是仍然是检测对主机的攻击，对于协同攻击和多域联合攻击没有检测的能力。另外，这方面的研究仍未停止，2001 年 10 月 23 日，SRI 发布了 eXpert-BSMTM for Solaris Version 1.4，据称是当时最先进的基于主机的 IDS。具有可升级、强大的事件分析能力、通用性能好，以及可插的组件等特性。

1.1.3　网络 IDS 研究

1990 年出现的网络安全监视器(Network Security Monitor，NSM)，是 UCD(Carlifornia 大学的 Davis 分校)设计的面向局域网的 IDS，NSM 被设计用来分析来自以太局域网的数据及连接到该网的数据。这个系统的重要贡献是首次使用网络数据包作为审计数据源，提出基于网络的 IDS 的概念。1991 年，NADIR(Network Anomaly Detection and Intrusion Reporter)与

DIDS(Distribute Intrusion Detection System)提出收集和合并来自多个主机的审计信息,来检测针对多个主机的协同攻击。需要指出的是,网络 IDS 的研究方法有两种:一是分析各主机的审计数据,并分析各主机审计数据之间的关系;二是分析网络数据包。

1994 年,美国空军密码支持中心(Cryptological Support Center)的一群研究人员创建了一个健壮的网络入侵检测系统 ASIM,该系统被广泛应用于美国空军,为了将网络入侵检测技术商业化,他们成立了一个商业公司 Wheelgroup。

1996 年,UCD(Carlifornia 大学的 Davis 分校)的计算机安全(Computer Security)实验室,以开发广域网上的入侵检测系统为目的,开发了 GrIDS。目标是使受保护的网络规模达到成千上万,甚至上百万,并且还保护路由器、域名服务器等。

1997 年,Cisco 公司兼并了 Wheelgroup,并开始将网络入侵检测整合到 Cisco 路由器中。同时,Internet 安全系统公司(ISS)发布了 RealSecure,这是一个被广泛使用的、用于 Windows NT 的网络入侵检测系统。从此,网络入侵检测革命的序幕被拉开了。

从 1996 年到 1999 年,SRI 开始 EMERALD(Event Monitoring Enabling Response to Anomalous Live Disturbances)的研究,它是 NIDES 的后继者,具有分布式可升级的特点,用于在大型网络中探测恶意入侵活动(包括对网站的入侵),高度分布,自动响应。并在以下几方面进行了扩展:可以进行基于网络的分析;增强互操作性;使分布式计算环境的集成更容易。根据 SRI 给出的报告,在 1999 年,美国政府发起的网络安全检测系统竞赛上,EMERALD 在 8 项性能测试中有 7 项位列榜首。

1.1.4 主机和网络入侵检测的集成

1990 年以前,大部分入侵检测系统都是基于主机的,它们对于活动性的检查局限于操作系统审计数据及其他以主机为中心的信息源。在 1.1.3 中提到,1990 年出现的 NSM 是面向局域网的 IDS,它把入侵检测扩展到了网络环境中。此时,由于 Internet 的发展及通信和网络带宽的增加,系统的互联性已经有了显著提高,导致人们对计算机安全的关注程度也显著增加。1988 年的 Internet 蠕虫事件使人们对计算机安全的关注达到了令人激动的程度,同时增加了对商业界和学术界的研究资助。分布式入侵检测系统(DIDS)最早试图把基于主机的方法和网络监视方法集成在一起。

DIDS 的开发是一个大规模的合作开发,参与方有美国空军密码支持中心、lawrence、Livermor 国家实验室、加利福尼亚大学 Davis 分校和 Haystack 实验室。这项研究由美国空军、国家安全部和国家能源部资助。它是将主机入侵检测和网络入侵检测的能力集成的第一次尝试,以便于一个集中式的安全管理小组能够跟踪安全侵犯和网络间的入侵。

DIDS 的最初概念是采用集中式控制技术,向 DIDS 中心控制器发送报告。DIDS 的结构如图 1-2 所示。

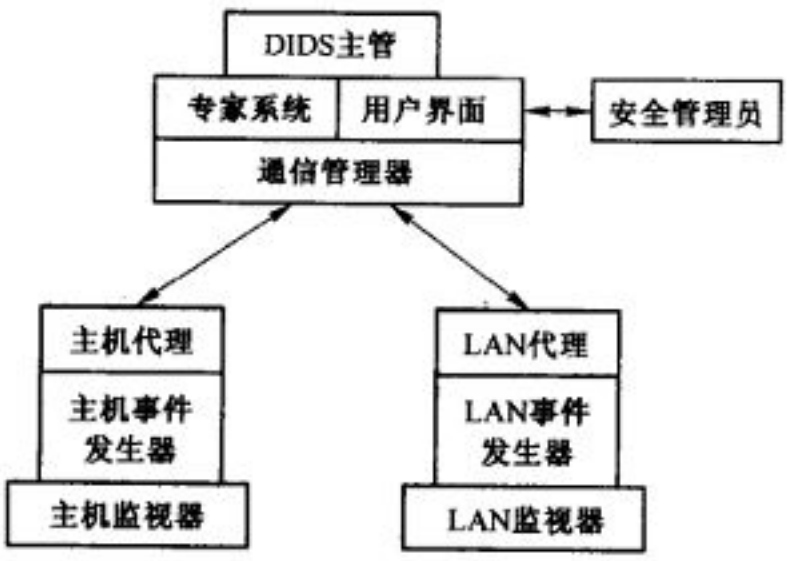

图 1-2 DIDS 结构

DIDS 解决了在大型网络互联中的一个棘手问题，即在网络环境下跟踪网络用户和文件。该项功能非常关键，这是因为：第一，网络入侵者通常会利用不同计算机系统的互联性来隐藏自己的真实身份和地址。实际上，一些入侵者发起的分布式攻击是在每个阶段从不同系统发起攻击的组合结果。第二，对付网络攻击的最有效的方法是发现对攻击负责的人，收集他进行攻击的证据，然后借助执法力量和法律过程来起诉他。系统允许用户在该环境中通过自动跨越被监视的网络跟踪和得到用户身份的相关信息来处理这个问题。DIDS 是第一个具有这个能力的入侵检测系统。

例如，假设攻击者通过“网络跳板”穿越系统，通过一个路由攻击受 DIDS 保护的支付服务器。DIDS 可以通过跟踪攻击者的身份，发现攻击路径节点上的各个用户的身份，例如是主机 1 上的 A、主机 2 上的 B 和主机 3 上的 C。此时，根据 DIDS 的结果，就可以派出调查员调查实际上坐在主机 1 的和 A 相关的终端前的那个人。

DIDS 解决的另一个问题是如何从发生在系统不同抽象层次的事件中发现相关数据或事件。这类信息要求要理解它们对整个网络的影响，DIDS 用一个 6 层的入侵检测模型提取数据相关性，每层代表了对数据的一次变换结果。

此外，还有许多系统，在此不一一赘述了。这些系统的分类如表 1-2 所示。

表 1-2　IDS 简单分类

异　常	自学习	非时间序列	规则模型	W&S
			基于统计	IDES, NIDES, EMERALD, Haystack
		时间序列	ANN	Haperview
	预编程的	描述统计	简单统计	MIDAS, NADIR, Haystack
			基于规则	NSM
			门限	ComputerWatch
		Default deny	状态序列模式	DPEM, JANUS, Bro
特征	预编程的	状态模式	状态转换	USTAT
			Petri 网	IDIOT
		专家系统	NIDES, EMERALD, MIDAS-direct, DIDS, MIDAS	
		字符串匹配	NSM	
		基于规则	NADIR, ASAX, Bro, Haystack	
自动特征	自学习	自动特征选取	Ripper	

1.2　入侵检测的基本概念

上节介绍了入侵检测的诞生及其大致的发展历程，为了使读者对入侵检测有一个清晰的了解，本节将介绍入侵检测的一些基本概念，包括入侵检测的概念、作用以及研究入侵检测的

必要性。

1.2.1 入侵检测的概念

入侵,是指任何试图危及计算机资源的完整性、机密性或可用性的行为。而入侵检测是对入侵行为的发觉。它通过从计算机网络或系统中的若干关键点收集信息,并对这些信息进行分析,从而发现网络或系统中是否有违反安全策略的行为和遭到攻击的迹象。进行入侵检测的软件与硬件的组合便是入侵检测系统(简称 IDS)。入侵检测是防火墙的合理补充,帮助系统对付网络攻击,它扩展了系统管理员的安全管理能力(包括安全审计、监视、进攻识别和响应),提高了信息安全基础结构的完整性。入侵检测被认为是防火墙之后的第二道安全闸门,在不影响网络性能的情况下能对网络进行监测,从而提供对内部攻击、外部攻击和误操作的实时保护。

网络入侵检测系统(IDS)是一项很新的网络安全技术,目前已经受到各界的广泛关注,它的出现是对原有安全系统的一个重要补充。入侵检测系统收集计算机系统和网络的信息,并对这些信息加以分析,对保护的系统进行安全审计、监控、攻击识别并作出实时的反应。

1.2.2 入侵检测的作用

形象地说,入侵检测系统就是网络摄像机,能够捕获并记录网络上的所有数据,同时它也是智能摄像机,能够分析网络数据并提炼出可疑的、异常的网络数据,它还是 X 光摄像机,能够穿透一些巧妙的伪装,抓住实际的内容。此外,它还是保安员的摄像机,能够对入侵行为自动地进行反击,如阻断连接。

在网络安全体系中,入侵检测系统是惟一一个通过数据和行为模式判断其是否有效的系统,如图 1-3 所示,防火墙就像一道门,可以阻止一类人群的进入,但无法阻止同一类人群中的破坏分子,也不能阻止内部的破坏分子;访问控制系统可以不让低级权限的人做越权工作,但无法保证高级权限的人做破坏工作,也无法阻止低级权限的人通过非法行为获得高级权限;漏洞扫描系统可以发现系统和网络存在的漏洞,但无法对系统进行实时扫描。

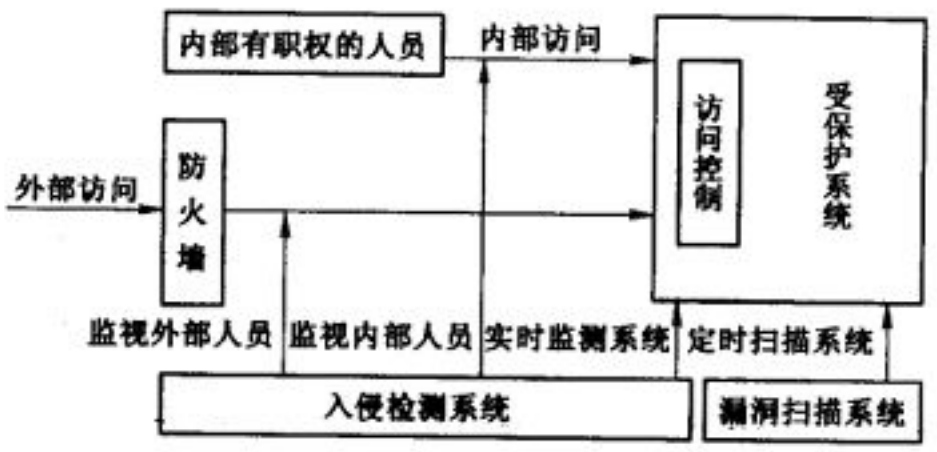

图 1-3 入侵检测的作用

入侵检测系统的作用和功能如下:

- 监控、分析用户和系统的活动。
- 审计系统的配置和弱点。

- 评估关键系统和数据文件的完整性。
- 识别攻击的活动模式。
- 对异常活动进行统计分析。
- 操作系统审计跟踪管理,识别违反政策的用户活动。

入侵检测系统的优点如下:

- 提高信息安全构造的其他部分的完整性。
- 提高系统的监控。
- 从入口点到出口点跟踪用户的活动。
- 识别和汇报数据文件的变化。
- 侦测并纠正系统配置错误。
- 识别特殊攻击类型,并向管理人员发出警报,进行防御。

入侵检测系统的缺点如下:

- 不能弥补差的认证机制。
- 如果没有人的干预,不能管理攻击调查。
- 不能知道安全策略的内容。
- 不能弥补网络协议上的弱点。
- 不能弥补系统提供质量或完整性的问题。
- 不能分析一个堵塞的网络。
- 不能处理有关 packet-level 的攻击。

对一个成功的入侵检测系统来讲,它不但可以使系统管理员时刻了解网络系统(包括程序、文件和硬件设备等)的任何变更,还能给网络安全策略的制订提供指南。更为重要的是,它的管理配置应该简单,从而使非专业人员非常容易地获得网络安全。而且,入侵检测的规模还应根据网络威胁、系统构造和安全需求的改变而改变。入侵检测系统在发现入侵后,会及时作出响应,包括切断网络连接、记录事件和报警等。

1.2.3　研究入侵检测的必要性

计算机网络安全应提供保密性、完整性以及抵抗拒绝服务的能力,但是由于联网用户的增加,越来越多的系统受到攻击。入侵者利用操作系统或者应用程序的缺陷企图破坏系统。为了对付这些攻击企图,可以要求所有的用户确认并验证自己的身份,并使用严格的访问控制机制,还可以用各种密码学方法对数据提供保护,但是这并不完全可行。另一种对付破坏系统企图的理想方法是建立一个完全安全的系统。但这样的话,就要求所有的用户能识别和认证自己,还要采用各种各样的加密技术和强访问控制策略来保护数据。而从实际上看,这根本是不可能的。这里有以下几方面的原因。

1) 在实践当中,建立完全安全系统根本是不可能的。Miller 给出一份有关现今流行操作系统和应用程序的研究报告,指出软件中不可能没有缺陷。另外,设计和实现一个整体安全系统相当困难。

2) 要将所有已安装的具有安全缺陷的系统转换成安全系统需要相当长的时间。

3) 如果口令是弱口令并且已经被破解,那么访问控制措施不能够阻止受到危害的授权用

户的信息丢失或者破坏。

4）静态安全措施不足以保护安全对象属性。通常，在一个系统中，担保安全特性的静态方法可能过于简单不充分，或者过度地限制用户。例如，静态技术未必能阻止违背安全策略造成浏览数据文件；而强制访问控制仅允许用户访问具有合适的通道的数据，这样就造成系统使用麻烦。因此，一种动态的方法如行为跟踪对检测和尽可能阻止安全突破是必要的。

5）加密技术方法本身存在的一定问题。

6）安全系统易受内部用户滥用特权的攻击。

7）安全访问控制等级和用户的使用效率成反比，访问控制和保护模型本身存在一定的问题。

8）在软件工程中存在软件测试不充足、软件生命周期缩短、大型软件复杂性等难解问题，工程领域的困难复杂性，使得软件不可能无错误，而系统软件容错恰恰是安全的弱点。

9）修补系统软件缺陷不能令人满意。由于修补系统软件缺陷，计算机系统不安全状态将持续相当长一段时间。

基于上述几类问题的解决难度，一个实用的方法是建立比较容易实现的安全系统，同时按照一定的安全策略建立相应的安全辅助系统。入侵检测系统就是这样一类系统，现在安全软件的开发方式基本上就是按照这个思路进行的。就目前系统安全状况而言，系统存在被攻击的可能性。但是，如果系统遭到了攻击，只要尽可能地检测到，甚至是实时地检测到，然后再采取适当的处理措施。入侵检测系统一般不是采取预防的措施以防止入侵事件的发生，入侵检测作为安全技术其主要目的有：识别入侵者；识别入侵行为；检测和监视已成功的安全突破；为对抗入侵及时提供重要信息，阻止事件的发生和事态的扩大。从这个角度看待安全问题，入侵检测非常必要，它将有效弥补传统安全保护措施的不足。

1.3 练习题

简答题

1. 简述分布式入侵检测系统(DIDS)是如何把基于主机的入侵检测方法和基于网络的入侵检测方法集成在一起的。

2. 简述入侵检测作用体现在哪些方面。

3. 简述研究入侵检测的必要性。

第 2 章　入侵检测系统

本章导读：

本章主要介绍关于入侵检测系统的相关知识。首先介绍了入侵检测系统的基本模型，包括通用入侵检测模型、层次化的入侵检测模型和 SNMP-IDSM 模型；接着介绍了入侵检测系统的工作模式和分类方法；然后根据入侵检测系统的类型详细分析了入侵检测系统的数据源；最后，简要介绍了入侵检测系统的部署方式。

2.1　入侵检测系统的基本模型

在入侵检测系统的发展历程中，大致经历了集中式、层次式和集成式三个阶段，代表这三个阶段的入侵检测系统的基本模型分别是通用入侵检测模型（Denning 模型）、层次化入侵检测模型（IDM）和管理式入侵检测模型（SNMP-IDSM），本节将介绍这三个基本模型。

2.1.1　通用入侵检测模型

Denning 于 1987 年最早提出一个通用的入侵检测系统模型（如图 2-1 所示）。

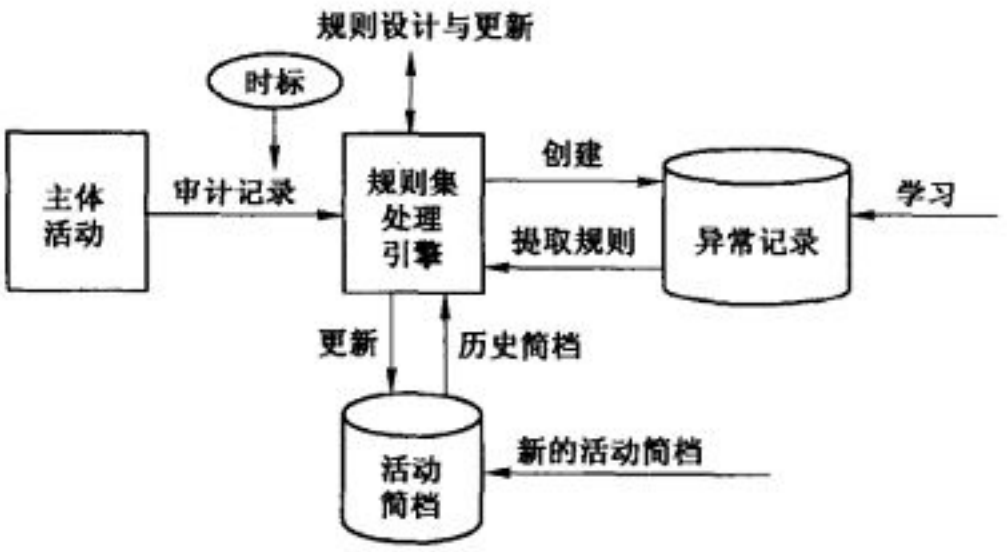

图 2-1　通用入侵检测系统模型

该模型由以下 6 个部分构成：

（1）主体（Subjects）

主体是指系统操作中的主动发起者，是在目标系统上活动的实体，例如计算机操作系统的进程、网络的服务连接等。

（2）对象（Objects）

对象指系统所管理的资源，如文件、设备、命令等。

（3）审计记录（Audit records）

审计记录是指主体对对象实施操作时，系统产生的数据，如用户注册、命令执行和文件访问等。审计记录是格式为＜Subject，Action，Object，Exception-Condition，Resource-Usage，Time-Stamp＞的六元组。其中：

- Subject(主体)，是活动(Action)的发起者。
- Action(活动)，是主体对目标实施的操作，对操作系统而言，这些操作包括读、写、登录、退出等。
- Object(对象)，是活动的承受者。
- Exception-Condition(异常条件)，是指系统对主体的该活动的异常报告，如违反系统读写权限。
- Resource-Usage(资源使用状况)，是系统的资源消耗情况，如 CPU、内存使用率等。
- Time-Stamp(时标)，是活动发生的时间。

(4) 活动简档(Activity Profile)

活动简档用以保存主体正常活动的有关信息，具体实现依赖于检测方法，在统计方法中从事件数量、频度、资源消耗等方面度量，可以使用方差、马尔可夫模型等方法实现。活动简档定义了以下三种类型的随机变量：

- 事件计数器(Event Counter)，简单地记录特定事件的发生次数。
- 间隔计时器(Interval Timer)，记录特定事件此次发生和上次发生之间的时间间隔。
- 资源计量器(Resource Measure)，记录某个时间内特定动作所消耗的资源量。

活动简档的格式为：＜Variable-name，Action-pattern，Exception-pattern，Resource-usage-pattern，Period，Variable-type，Threshold，Subject-pattern，Object-pattern，Value＞。其中：

- Variable-name(变量名)，用来识别活动简档的标志。
- Action-pattern(活动模式)，用来匹配审计记录中的零个或多个活动的模式。
- Exception-pattern(异常模式)，用来匹配审计记录中的异常情况的模式。
- Resource-usage-pattern(资源使用模式)，用来匹配审计记录中的资源使用的模式。
- Period，测量的间隔时间或者取样时间。
- Variable-type，一种抽象的数据类型，用来定义一种特定的变量和统计模型。
- Threshold(阈值)，统计测试中一种表示异常的参数值。
- Subject-pattern(主题模式)，用来匹配审计记录中主体的模式，识别活动简档的标志。
- Object-pattern(对象模式)，用来匹配审计记录中的对象的模式，识别活动简档的标志。
- Value，当前观测值和统计模型所用的参数值，例如平均值和标准差模型中这些参数可能是变量和或者是变量的平方和。

(5) 异常记录(Anomaly Record)

异常记录用以表示异常事件的发生情况，格式为＜Event，Time-stamp，Profile＞，其中：

- Event，指明导致异常的事件，例如审计数据。
- Time-stamp，产生异常事件的时间戳。
- Profile，检测到异常事件的活动简档。

(6) 活动规则

活动规则指明当一个审计记录或异常记录产生时应采取的动作。规则集是检查入侵是否

发生的处理引擎，结合活动简档用专家系统或统计方法等分析接收到的审计记录，调整内部规则或统计信息，在判断有入侵发生时采取相应的措施。规则由条件和动作两部分组成，共有四种类型的规则，分别为：

- 审计记录规则(Audit-record rules)：触发新生成审计记录和动态的活动简档之间的匹配以及更新活动简档和检测异常行为。
- 定期活动更新规则(Periodic-activity-update rules)：定期触发动态活动简档中的匹配以及更新活动简档和检测异常行为。
- 异常记录规则(Anomaly-record rules)：触发异常事件的产生，并将异常情况报告给安全管理员。
- 定期异常分析规则(Periodic-anomaly-analysis rules)：定期触发产生当前的安全状态报告。

Denning 模型实际上是一个基于规则的模式匹配系统，不是所有的 IDS 都能够完全符合该模型。Denning 模型的最大缺点在于它没有包含已知系统漏洞或攻击方法的知识，而这些知识在许多情况下是非常有用的信息。

2.1.2　IDM 模型

Steven Snapp 等人在设计和开发分布式入侵检测系统 DIDS 时，提出一个层次化的入侵检测模型，简称 IDM。该模型将入侵检测分为六个层次，分别为：数据(data)、事件(event)、主体(subject)、上下文(context)、威胁(thread)、安全状态(security state)。

IDM 模型给出了在推断网络中的计算机受攻击时数据的抽象过程。也就是说，它给出了将分散的原始数据转换为高层次有关入侵和被监测环境的全部安全假设过程。通过把收集到的分散数据进行加工抽象和数据关联操作，IDM 构造了一台虚拟的机器环境，这台机器由所有相连的主机和网络组成。将分布式系统看作一台虚拟计算机的观点简化了跨越单机入侵行为的识别。IDM 也应用于只有单台计算机的小型网络。IDM 六个层次的详细情况如下：

(1) 第一层(数据)

该层包括主机操作系统的审计记录、LAN 监视器结果和第三方的审计软件包提供的数据。在该层中，刻画客体的语法和语义与数据来源是相关联的，主机或网络上的所有操作都可以用这样的客体表示出来。

(2) 第二层(事件)

该层处理的客体是对第一层客体的扩充，该层的客体称为事件。实际描述第一层的客体内容所表示的含义和固有特征性质。用来说明事件的数据域有两个，即动作(action)和领域(domain)。动作刻画了审计记录的动态特征，而领域给出了审计记录的对象的特征。很多情况下，对象是指文件或设备，但它们的领域判断要根据对象的特征或它所在文件系统的位置来确定。由于进程也是审计记录的对象，它们也可以归到某个领域，这时就要看进程的功能。事件的动作包括会话开始、会话结束、读文件或设备、写文件或设备、进程执行、进程结束、创建文件或设备、删除文件或设备、移动文件或设备、改变权限、改变用户号等。事件的领域包括标签、认证、审计、网络、系统、系统信息、用户信息、应用工具、拥有者和非拥有者等。

(3) 第三层(主体)

该层是一个惟一标识号,用来鉴别在网络中跨越多台主机使用的用户。

(4) 第四层(上下文)

该层用来说明事件发生时所处的环境,或者给出事件产生的背景。上下文分为时间型和空间型两类。若一个在用户正常工作时间内不出现的操作在下班时出现,则这个操作很值得怀疑,这就是时间型上下文的例子。另外,事件发生的时间顺序也常能用作检测入侵,例如一个用户频繁注册失败就可能表明入侵正在发生。IDM 要选取某个时间为参考点,然后利用相关的事件信息来检测入侵。空间型上下文说明了事件的来源与入侵行为的相关性,事件与特别的用户或者一台主机相关联。例如,我们关心一个用户从低安全级别计算机向高安全级别计算机的转移操作,而反方向则不太重要。这样,事件上下文使得可以对多个事件进行相关性入侵检测。

(5) 第五层(威胁)

该层考虑事件对网络和主机构成的威胁。当把事件及其上下文结合起来分析时,就能够发现存在的威胁。可以根据滥用的特征和对象对威胁类型进行划分,也就是说,入侵者做了什么和入侵的对象是什么。滥用分为攻击、误用和可疑等三种操作。攻击表明机器的状态发生了改变,误用则表示越权行为,而可疑只是入侵检测感兴趣的事件,但是不与安全策略冲突。滥用的目标划分成系统对象或是用户对象、被动对象或是主动对象。用户对象是指没有权限的用户或者是用户对象存放在没有权限的目录。系统对象则是用户对象的补集。被动对象是文件,而主动对象是运行的进程。

(6) 第六层(安全状态)

IDM 的最高层用 1～100 的数字值来表示网络的安全状态,数字越大,网络的安全越低。实际上,可以将网络安全的数字值看作是系统中所有主体产生威胁的函数。尽管这种表示系统安全状态的方法会丢失部分信息,但是可以使安全管理员对网络系统的安全状态有一个整体印象。DIDS 实现 IDM 模型时,采用一个内部数据库保存各个层次的信息,安全管理员可以根据需要查询详细的相关信息。

2.1.3 SNMP-IDSM 模型

近年来,随着网络技术的飞速发展,网络攻击手段也越来越复杂,攻击者大都是通过合作的方式来攻击某个目标系统,而单独的 IDS 难以发现这种类型的入侵行为。然而,如果 IDS 系统也能够像攻击者那样合作,就有可能检测到入侵者。这样就要有一种公共的语言和统一的数据表达格式,能够让 IDS 系统之间顺利交换信息,从而实现分布式协同检测。但是,相关事件在不同层面上的抽象表示也是一个很复杂的问题。基于这样的因素,北卡罗莱那州立大学的 Felix Wu 等人从网络管理的角度考虑 IDS 的模型,提出了基于 SNMP 的 IDS 模型,简称 SNMP-IDSM。

SNMP-IDSM 以 SNMP 为公共语言来实现 IDS 系统之间的消息交换和协同检测,它定义了 IDS-MIB,使得原始事件和抽象事件之间关系明确,并且易于扩展。SNMP-IDSM 的工作原理如图 2-2 所示。IDS B 负责监视主机 B 和请求最新的 IDS 事件,主机 A 的 IDS A 观察到一个来自主机 B 的攻击企图,然后 IDS A 与 IDS B 联系,IDS B 响应 IDS A 的请求,IDS B 半小时前发现有人扫描主机 B,这样,某个用户的异常活动事件被 IDS B 发布。IDS A 怀疑主机 B 受

到了攻击。为了验证和寻找攻击者的来源，IDS A 使用 MIB 脚本发送一些代码给 IDS B。这些代码的功能类似于“netstat、lsof”等，它们能够搜集主机 B 的网络活动和用户活动的信息。最后，这些代码的执行结果表明用户 X 在某个时候攻击主机 A，而且，IDS A 进一步得知用户 X 来自于主机 C。这样，IDS A 和 IDS C 联系，要求主机 C 向 IDS A 报告入侵事件。

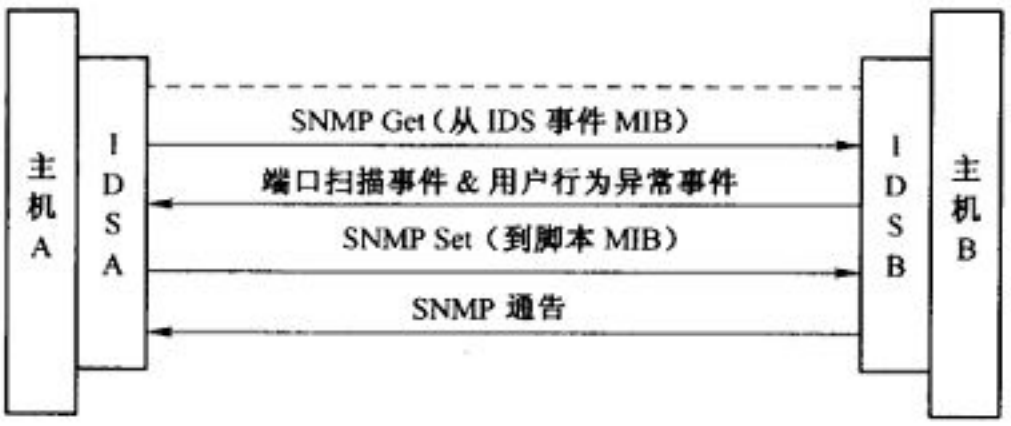

图 2-2　IDS A 与 IDS B 消息交换示意图

一般来说，攻击者在一次入侵过程中通常会采取以下步骤：

- 使用端口扫描、操作系统检测或者其他黑客工具收集目标的有关信息。
- 寻找系统的漏洞并利用这些漏洞，例如 sendmail 的错误、匿名 FTP 的误配置或者 X 服务器授权给任何人访问。一些攻击企图失败而被记录下来，另外一些攻击企图则可能成功实施了。
- 如果攻击成功，入侵者就会清除日志记录或者隐藏自己而不被其他人观察到。
- 安装后门，如 rootkit、木马或网络嗅探器等。
- 使用已攻破的系统作为跳板入侵其他主机，例如用窃听口令攻击相邻的主机或者搜索主机间非安全信任关系等。

SNMP-IDSM 根据上述的攻击原理，采用五元组来描述攻击事件，该五元组如下所示：<WHERE，WHEN，WHO，WHAT，HOW>。其中各个字段的含义如下：

- WHERE：描述产生攻击的位置，包括目标所在地以及在什么地方观察到事件发生。
- WHEN：事件的时间戳，用来说明事件的起始时间、终止时间、信息频度或发生的次数。
- WHO：表明 IDS 观察到的事件，如果可能的话，记录哪个用户或进程触发事件。
- WHAT：记录详细信息，例如协议类型、协议说明数据和包的内容。
- HOW：用来连接原始事件和抽象事件。

SNMP-IDSM 定义了用来描述入侵事件的管理信息库 MIB，并将入侵事件分为原始事件(Raw Event)和抽象事件(Abstract Event)两层结构。原始事件指的是引起安全状态迁移的事件或者是表示单个变量偏移的事件，而抽象事件是指分析原始事件所产生的事件。原始事件和抽象事件的信息都用四元组<WHERE，WHEN，WHO，WHAT>来描述。

2.2　入侵检测系统的工作模式

无论对于什么样的入侵检测系统，其工作模式都体现为以下四个步骤：

- 从系统的不同环节收集信息。

- 分析该信息,试图寻找入侵活动的特征。
- 自动对检测到的行为作出响应。
- 记录并报告检测过程及结果。

一个典型的入侵检测系统从功能上可以分为三个组成部分:感应器(sensor)、分析器(analyzer)和管理器(manager)。如图 2-3 所示。

图 2-3 入侵检测系统的功能结构

其中,感应器负责收集信息。其信息源可以是系统中可能包含入侵细节的任何部分,其中比较典型的信息源有网络数据包、log 文件和系统调用的记录等。感应器收集这些信息并将其发送给分析器。

分析器从许多感应器接收信息,并对其进行分析以决定是否有入侵行为发生。如果有入侵行为发生,分析器将提供关于入侵的具体细节,并提供可能采取的对策。一个入侵检测系统通常也可以对所检测到的入侵行为采取相应的措施进行反击,包括在防火墙处丢弃可疑的数据包,当用户表现出不正常行为时拒绝其进行访问,以及向其他同时受到攻击的主机发出警报等。

管理器通常也被称为用户控制台,它以一种可视的方式向用户提供收集到的各种数据及相应分析结果,用户可以通过管理器对入侵检测系统进行配置,设定各种系统的参数,从而对入侵行为进行检测及相应措施进行管理。

2.3 入侵检测系统的分类

通过对现有的入侵检测系统和技术的研究,可以从下面几个方面对入侵检测系统进行分类:

(1) 根据目标系统的类型

根据目标系统类型的不同可以将入侵检测系统分为如下两类:

- 基于主机(Host-Based)的入侵检测系统。通常,基于主机的入侵检测系统可监测系统、事件和操作系统下的安全记录以及系统记录。当有文件发生变化时,入侵检测系统将新的记录条目与攻击标记相比较,看它们是否匹配。如果匹配,系统就会向管理员报警,以便采取措施。
- 基于网络(Network-Based)的入侵检测系统。基于网络的入侵检测系统使用原始网络包作为数据源。该系统通常利用一个运行在混杂模式下的网络适配器来实时监视并分析通过网络的所有通信业务。

(2) 根据入侵检测系统分析的数据来源

入侵检测系统分析的数据可以是主机系统日志、网络数据包、应用程序的日志、防火墙报警日志以及其他入侵检测系统的报警信息等。据此可将入侵检测系统分为基于不同分析数据源的入侵检测系统。

(3) 根据入侵检测方法

根据入侵检测分析方法的不同可将入侵检测系统分为如下两类:

- 异常入侵检测系统。异常入侵检测系统利用被监控系统正常行为的信息作为检测系统中入侵、异常活动的依据。

- 误用入侵检测系统。误用入侵检测系统根据已知入侵攻击的信息(知识、模式等)来检测系统中的入侵和攻击。

(4) 根据检测系统对入侵攻击的响应方式

根据检测系统对入侵攻击的响应方式的不同可将入侵检测系统分为如下两类:

- 主动的入侵检测系统。主动的入侵检测系统在检测出入侵后,可自动地对目标系统中的漏洞采取修补、强制可疑用户(可能的入侵者)退出系统以及关闭相关服务等对策和响应措施。
- 被动的入侵检测系统。被动的入侵检测系统在检测出对系统的入侵攻击后只是产生报警信息通知系统安全管理员,至于之后的处理工作则由系统管理员完成。

(5) 根据系统各个模块运行的分布方式

根据系统各个模块运行的分布方式的不同,可将入侵检测系统分为如下两类:

- 集中式入侵检测系统。系统的各个模块包括数据的收集与分析以及响应都集中在一台主机上运行,这种方式适用于网络环境比较简单的情况。
- 分布式入侵检测系统。系统的各个模块分布在网络中不同的计算机、设备上,一般来说分布性主要体现在数据收集模块上,如果网络环境比较复杂、数据量比较大,那么数据分析模块也会分布,一般是按照层次性的原则进行组织。

当前,众多入侵检测系统和技术,基本上都是根据检测方法、目标系统、信息数据源来设计的,下面将针对这些方法进行详述。

2.4　入侵检测系统的数据源

入侵检测系统中用于分析检测的信息主要来源于系统主机的日志记录、网络数据包、系统针对应用程序的日志数据以及其他入侵检测系统或系统监控系统的报警信息。

最初的入侵检测系统针对的目标系统大多是主机系统,所有的系统用户对于系统来说都是本地的,很少有来自外界的攻击,这使得入侵检测系统只需要分析由主机系统提供的审计信息就可以完成检测系统入侵的任务。而在分布式环境中,用户可以从一台机器跳到另一台机器,而且可能在跳动过程中采用不同的用户标识,这样他们就可以通过不同的机器对网络进行分散攻击。因此,在工作站上的本地入侵检测系统必须和网络中其他工作站上的入侵检测系统交换信息。这些通过网络交换的信息可以是各自的原始审计数据(如 ASAX、Stalker),也可以是本地检测系统的报警结果。但这两种方案的代价都很大:传递大量的审计信息对网络的带宽要求很高,否则极易造成网络阻塞;若在本地分析以提供局部的报警结果,则又严重影响工作站正常工作的性能。

2.4.1　基于主机的数据源

审计数据是收集一个给定机器用户活动信息的惟一方法。但是,当系统受到攻击时,系统的审计数据很有可能被修改。这就要求基于主机的入侵检测系统必须满足一个重要的实时性条件:检测系统必须在攻击者接管机器并暗中破坏系统审计数据或入侵检测系统之前完成对审计数据的分析、产生报警并采取相应的措施。

(1) 系统运行状态信息

所有的操作系统都提供了一些系统命令来获取系统运行情况。在 UNIX 环境中,这类命令有:ps、pstat、vmstat、getrlimit 等。这些命令直接检查系统内核的存储区,所以它们能够提供相当精确的关于系统事件的关键信息。但是由于这些命令不能提供结构化的方法来收集或存储对应的审计信息,所以很难满足入侵检测系统需要连续进行审计数据收集的需求。

(2) 系统记帐信息

记帐(Accounting)是获取系统行为信息的最古老、普遍的方法。在网络设备、主机系统以及 UNIX 工作站中都使用了记帐系统,用于提供系统用户使用共享资源(例如,处理机时间、内存、磁盘或网络的使用等)的信息,以便向用户收费。记帐系统的广泛应用使得在设计入侵检测原型时可以采用它做为系统审计数据源。

在 UNIX 环境中,记帐系统是一个通用性的信息源,而且具有以下特点:

- 在所有的 UNIX 系统中,记帐信息记录的格式都是统一的。
- 记帐信息进行了压缩以节省磁盘空间。
- 记帐信息记录的处理开销非常小。
- 记帐系统可与现代操作系统很好地集成,且很容易建立和使用。

但是,记帐系统也有一系列的缺点,使它不能可靠地做为入侵检测系统的分析数据源:

- 记帐文件有时会存放在用户可操作的磁盘分区内,这时用户只需简单地填充该分区,使其使用率达到 90% 以上,那么系统的记帐就会停止。
- 记帐系统可以被打开或关闭,但是不能只对指定的用户进行记帐。
- 记帐信息缺乏精确的时戳。记录的系统命令不能按它们实际发出的时间排序,而命令序列在一些入侵检测系统中则是一种重要的检测信息。
- 缺乏精确的命令识别。在记帐记录中只存有用户发出命令名的前八个字符,而重要的路径信息和命令行参数则全丢掉了。这样,一些基于知识的入侵检测系统就不能有效地检测出“特洛伊木马”攻击。
- 缺乏系统守护程序的活动记录。记帐系统只能记录关于运行终止的信息,因而针对诸如 Sendmail 这类一直运行的系统守护程序则不会记录。
- 获取信息的时间太迟。记帐信息记录当应用终止时才能写入文件,这时入侵活动可能已经发生了。

基于以上原因,系统记帐信息从未在基于知识的入侵检测系统中使用过,而且在基于行为的入侵检测系统中也很少使用。在 Hyperview 的统计和神经网络模块中使用它也只是作为审计数据的一个补充而已。

(3) 系统日志(Syslog)

Syslog 是操作系统为系统应用提供的一项审计服务,这项服务在系统应用提供的文本串形式的信息前面添加应用运行的系统名和时戳信息,然后进行本地或远程归档处理。但是 Syslog 并不安全,因为据 CERT 的报告,一些 UNIX 的 Syslog 守护程序极易遭受缓冲区溢出性攻击。不过 Syslog 很容易使用,诸如 login、sendmail、nfs、http 等系统应用和网络服务,还有安全类工具(例如,sudo、klaxon 以及 TCP wrappers 等)都使用它作为自己的审计记录。但只有少数入侵检测系统采用 Syslog 守护程序提供的信息。

(4) C2 级安全性审计信息

系统的安全审计记录了系统中所有潜在的安全相关事件的信息。在 UNIX 系统中,审计系统记录了用户启动的所有进程执行的系统调用序列。和一个完整的系统调用序列比较,审计记录则进行了有限的抽象,其中没有出现上下文切换、内存分配、内部信号量以及连续的文件读的系统调用序列。这也是一个把审计事件映射为系统调用序列的直接方法。UNIX 的安全审计记录中包含了大量关于事件的信息:用于识别用户、组的详细信息(登录身份、用户相关的程序调用)、系统调用执行的参数(包含路径的文件名、命令行参数等)以及系统程序执行的返回值、错误码等。使用安全审计的主要优点有:

- 可以对用户的登录身份、真实身份、有效身份以及真实有效的所属组的标识进行强验证。
- 可以很容易地通过配置审计系统实现审计事件的分类。
- 可根据用户、类别、审计事件或系统调用的成功和失败获取详细的参数化信息。
- 当审计系统遇到一个错误状态(通常是磁盘空间耗尽)时,机器就会被关闭。

使用安全审计的主要缺点:

- 当需要进行详细监控时,会消耗大量系统资源,处理机性能可能会降低 20%,并且还需要大量本地磁盘空间对审计数据进行存储和归档。
- 通过填充审计系统的磁盘空间可造成拒绝服务攻击。
- 不同操作系统的审计记录格式和审计系统接口的异构性,使得从异构环境中获得的审计数据不仅数据量大且构成复杂。所以在利用安全审计数据进行检测时,也存在很大的困难。

由于 C2 级安全审计是目前惟一能够对信息系统中活动的详细信息进行可靠收集的机制,它在大多数的入侵检测系统原型以及检测工具中都做为主要的审计信息源。一些研究小组建议制定一个审计记录的通用格式并定义哪些信息必须包含在审计记录中,这项工作还处在研究之中。

2.4.2　基于网络的数据源

在当前商业入侵检测产品中,网络传输是最常见的数据源。在基于网络的入侵检测方法中,需要从网络上传输的网络通信流中采集信息。

基于网络的数据源流行的主要原因是通过网络监控获得信息的性能代价很低,这是因为当数据包通过网络时,监控器很容易读取它们。因此,运行监控器不影响网络上运行的其他系统的性能。

基于网络的数据源的另一个优点是,在网络上监控器对用户可以是透明的,因此降低了攻击者无需大量努力就能找到它并使之无效的可能性。由于监控系统需要的主要资源是存储空间,所以完全可以使用较旧的、较慢的系统对网络部分进行监控。

此外,网络监控器可以发现对基于主机系统来说不容易发现的某种攻击的证据。这些攻击包括基于非法格式包和各种拒绝服务攻击的网络攻击。

(1) SNMP 信息

简单网络管理协议(SNMP)的管理信息库(MIB)是一个用于网络管理的信息库。其中存

储有网络配置信息(路由表、地址、域名等)以及性能/记帐数据(不同网络接口和不同网络层的业务测量的计数器)。在 SecureNet 中曾利用 SNMP 版本 1 管理信息库中的计数器信息来作为基于行为的入侵检测系统的输入信息。一般在网络接口层检查这些计数器,这是因为网络接口主要用来区分信息是发送到网线还是通过回路接口发送回操作系统内部。另外,有些项目组在他们的安全工具研究中考虑使用了 SNMP 版本 3 的相关信息 。

(2) 网络通信包

网络嗅探器做为收集网络中发生事件信息的有效方法,常被攻击者用来截取网络数据包以获取有用的系统信息。目前多数攻击计算机系统的行为是通过网络进行的,通过监控、查看出入系统的网络数据包,来捕获口令或全部内容。这种方法是一种有效的攻入系统内部的方法。几乎所有的拒绝服务攻击都是基于网络的攻击,而且对它们的检测也只能借助于网络,因为基于主机的入侵检测系统靠审计系统不能获取关于网络数据传输的信息。入侵检测系统在利用网络通信包作为数据源时,如果入侵检测系统作为过滤路由器,直接利用模式匹配、签名分析或其他方法对 TCP 或 IP 报文的原始内容进行分析,那么分析的速度就会很快。但如果入侵检测系统作为一个应用网关来分析与应用程序或所用协议有关的每个数据报文时,对数据的分析就会更彻底,但开销很大。利用网络通信包做入侵检测系统的分析数据源,可以解决以下安全问题:

- 检测只能通过分析网络业务才能检测出来的网络攻击。例如,拒绝服务攻击。
- 不存在基于主机的入侵检测系统在网络环境下遇到的审计记录格式异构性的问题。TCP/IP 作为事实上的网络协议标准使得利用网络通信包的入侵检测系统不用考虑数据采集、分析时数据格式的异构性。
- 由于使用一个单独的机器进行信息的收集,因而这种数据收集、分析工具不会影响整个网络的处理性能。
- 某些工具可通过签名分析报文载荷内容或报文的头信息,来检测针对主机的攻击。

但这种方法也存在一些典型弱点:

- 当检测出入侵时,很难确定入侵者。因为在报文信息和发出命令的用户之间没有可靠的联系。
- 加密技术的应用使得对报文载荷的分析变得不可能,从而使这些检测工具失去大量有用的信息。

如果这些工具基于商用的操作系统来获取网络信息,由于商用的操作系统中的堆栈易于遭受拒绝服务攻击,所以建立在其上的入侵检测系统也就不可能避免遭受攻击。

2.4.3 应用程序日志文件

系统应用服务器化的趋势,使得应用程序的日志文件在入侵检测系统的分析数据源中具有相当重要的地位。与系统审计记录和网络通信包相比,使用应用程序日志文件具有以下三方面的优势:

- 精确性(Accuracy):对于 C2 审计数据和网络通信包,它们必须经过数据预处理,才能使入侵检测系统了解应用程序相关的信息。这种处理过程基于协议规范和应用程序接口(API)规范的解释,但是应用程序开发者的解释可能与入侵检测系统中的解释不一致,

从而造成入侵检测系统对安全信息的理解偏差。而直接从应用日志中提取信息，就可以尽量保证入侵检测系统获取安全信息的准确性。

- 完整性(Completeness)：使用 C2 审计数据或网络通信包时，为了重建应用层的会话，需要对多个审计调用或网络通信包进行重组(特别是在多主机系统中)。但却很难达到要求，即使最简单的重组需求(例如，通过匹配 HTTP 请求和响应来确定一个成功的请求)，用目前的工具也很难完成。而对应用程序日志文件来说，即使应用程序是一个运行在一组计算机上的分布式系统(诸如，Web 服务器、数据库服务器等)，它的日志文件也能包含所有的相关信息。另外，应用程序还能提供审计记录或网络包中没有的内部数据信息。
- 性能(Performance)：通过应用程序选择与安全相关的信息，使得系统的信息收集机制的开销远小于利用安全审计记录的情况。

虽然使用应用程序日志文件有以上的优点，但也有一些缺点：

- 只有当系统能够正常写应用程序日志文件时，才能够检测出对系统的攻击行为。如果对系统的攻击使系统不能记录应用程序的日志(在许多拒绝服务攻击中都会出现这种情况)，那么入侵检测系统将得不到检测所需的信息。
- 许多入侵攻击只是针对系统软件低层协议中的安全漏洞，诸如网络驱动程序、IP 协议等。而这些攻击行为不利用应用程序的代码，所以它们的攻击情况在应用程序的日志中看不出来，惟一能够看到的就是攻击的结果(诸如，系统被重启动)。

IBM 公司的 WebWatcher 是一个典型的利用应用程序日志文件的入侵检测工具，它通过实时地对 Web 服务器的日志进行监控来获取大量针对 Web 服务器攻击的详细信息，并据此进行检测。同样，可以设计监控数据库服务器的入侵检测工具。

2.4.4　其他入侵检测系统的报警信息

随着网络技术和分布式系统的发展，入侵检测系统也从针对主机系统转向针对网络、分布式系统。基于网络、分布式环境的检测系统为了覆盖较大的范围，一般采用分层的结构，有许多局部的入侵检测系统(可以是传统的基于主机的入侵检测系统)进行局部的检测，然后把局部检测结果汇报给上层的检测系统，而且各局部入侵检测系统也可采用其他局部入侵检测系统的结果做参考，以弥补不同检测机制的入侵检测系统的不足。因此，其他入侵检测系统的报警信息也是入侵检测系统的重要数据来源。典型的系统有 DIDS、GrIDS 等。其中，DIDS 把基于主机系统的 Haystack 和基于网络的 NSM 检测系统组合到一起，对它们进行控制，并利用它们的报警信息进一步分析检测。GrIDS 是一个基于图形分析的入侵检测系统，它能够检测出跨越大型网络基础设施的入侵攻击行为。检测时它把局部化的基于主机的或基于网络的入侵检测系统的检测结果按图形的结构形式汇集起来进行分析，这种对结果图的分析可以突出不同的入侵攻击的相关性。

2.5　入侵检测系统的部署

对于入侵检测系统来说，其类型不同、应用环境不同，部署方案也就会有所差别。对于基

于主机的入侵检测系统来说，它一般是用于保护关键主机或服务器，因此只要将它部署到这些关键主机或服务器中即可。但是对于基于网络的入侵检测系统来说，根据网络环境的不同，其部署方案也就会有所不同，各种网络环境千差万别，在此无法一一赘述，因此在本节中只考虑两种情况的网络环境，即网络中没有部署防火墙时的情况和网络中已经部署防火墙时的情况。

1. 网络中没有部署防火墙时

通常在网络中考虑安全时，首先考虑到的是在网络入口处安装防火墙进行过滤。但是在有些环境中由于某种原因可能无法部署防火墙。

在没有安装防火墙的情况下，网络入侵检测系统通常安装在网络入口处的交换机或者是集线器上，一边监听所有进出网络的数据包并进行相应的保护，如图 2-4 所示。在交换机环境下为了监听所有的数据包，通常利用交换机的端口镜像功能。具体的镜像配置方法各交换机厂商存在一些差异，需要向交换机厂商或者经销商咨询。

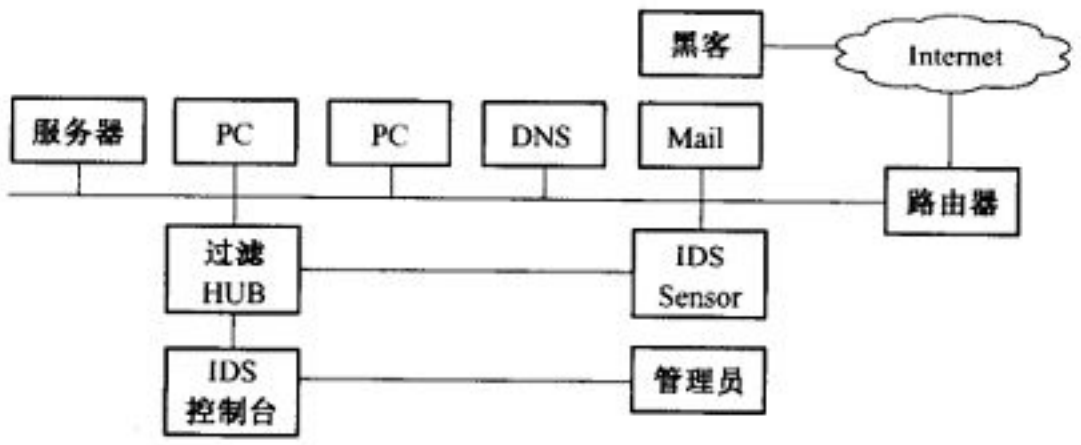

图 2-4 没有部署防火墙时入侵检测系统的部署情况

目前市面上的 4 层交换机除了具有端口镜像的功能之外，还提供 QoS、负载平衡等功能，部分提供商的产品提供防御部分 DOS(拒绝服务)攻击的能力，所以有利于提高整体安全性。

2. 网络中部署防火墙时

防火墙系统起防御来自外部网络的攻击的作用，在这时和入侵检测系统互相配合可以进行更有效的安全管理。

在这种情况下通常将入侵检测系统部署在防火墙之后，进行继防火墙一次过滤后的二次防御，如图 2-5 所示。

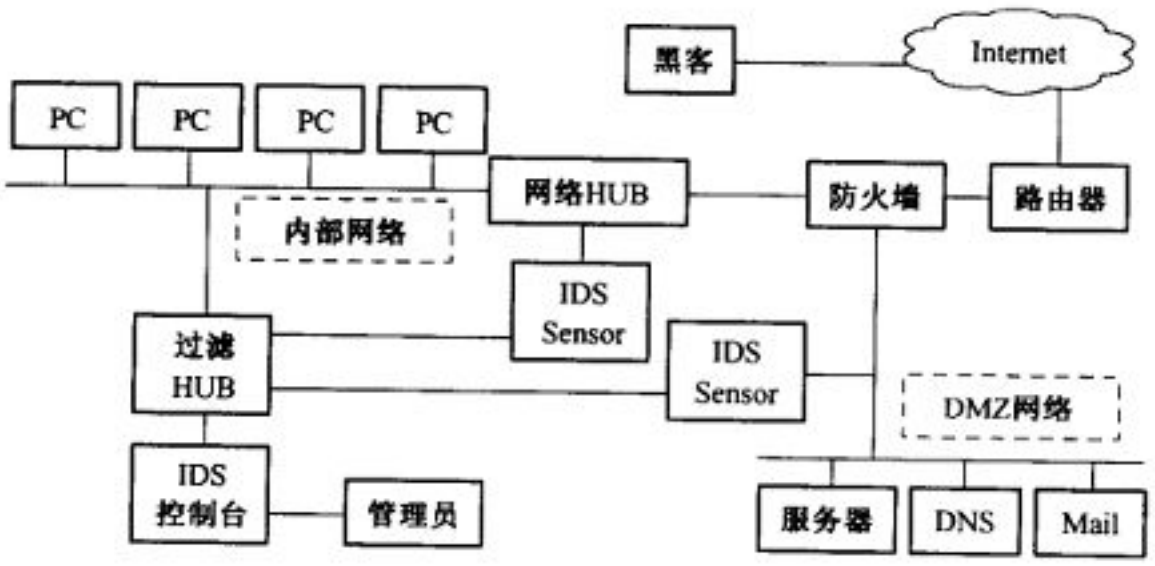

图 2-5 入侵检测系统部署在防火墙内部

但是在有些情况下，还需要考虑来自外部的针对防火墙本身的攻击行为。如果黑客觉察到防火墙的存在并功破防火墙的话，对内部网络来说是极其危险的。因此在高安全性要求的环境下应该在防火墙外部部署入侵检测产品，进行先于防火墙的一次检测、防御。这样用户可以预知那些恶意攻击防火墙的行为并及时采取相应的安全措施，以保证整个网络的安全性。

2.6　练习题

一、选择题

1. Denning 模型是一个基于(　　)的模式匹配系统。

A. 规则　　B. 模式

C. 异常行为　　D. 入侵发现

2. 异常入侵检测系统利用被监控系统(　　)的信息作为检测系统中入侵、异常活动的依据。

A. 日志记录　　B. 网络通信包

C. 正常行为　　D. 异常行为

3. 分布式入侵检测系统的各个模块分布在网络中不同的计算机、设备上，一般来说分布性主要体现在(　　)模块上。

A. 数据处理　　B. 数据收集

C. 数据分析　　D. 告警响应

4. (　　)不属于基于主机的数据源。

A. 系统日志　　B. 系统运行状态信息

C. SNMP 信息　　D. 系统记帐信息

5. 下面不是应用程序日志文件的优势的是(　　)。

A. 精确性　　B. 实时性

C. 完整性　　D. 性能

6. IDM 模型将入侵检测分为六个层次，下列不属于这六个层次中的选项是(　　)。

A. 数据　　B. 事件

C. 主体　　D. 安全管理

7. 一个典型的入侵检测系统从功能上可以分为三个组成部分，下列不属于这三个部分的选项是(　　)。

A. 感应器　　B. 分析器

C. 管理器　　D. 协调器

8. 一个基于网络的 IDS 应用程序利用(　　)来检测攻击。

A. 正确配置的 DNS　　B. 特征库

C. 攻击描述　　D. 信息包嗅探器

9. 下列不属于 IDS 部件的是(　　)。

A. 事件产生器　　B. 事件分析器

C. 事件存储器　　D. 事件数据库

10. 可以确保未授权用户无法对 NIDS 的应用程序的访问是(　　)。

A. 确保系统是物理安全的　　B. 在不用的时候卸载程序

C. 设置身份认证　　D. 对所有的通信加密

二、简答题

1. 简述 IDM 模型的工作原理。
2. 简述 SNMP-IDSM 模型的工作原理。
3. 分析入侵检测系统的工作模式。
4. 简述异常入侵检测系统和误用入侵检测系统的本质区别。
5. 简述防火墙对部署入侵检测系统的影响。

第3章　入侵检测技术

本章导读：

入侵检测涉及到很多技术的应用，而且目前这些技术还在不断发展之中。本章主要介绍入侵检测技术。首先，对入侵检测过程进行了分析；然后介绍了入侵分析的概念和入侵分析的模型；接下来介绍了入侵分析的方法，对异常入侵检测和误用入侵检测的相关分析方法进行了重点介绍和分析；接着分析了入侵检测中的各种告警与响应方式；最后对入侵追踪进行了比较详细的介绍。

3.1　入侵检测的过程

入侵检测的过程可以分为三个阶段：信息收集、信息分析以及告警与响应。本节将进行详细叙述。

3.1.1　信息收集

入侵检测的第一步是信息收集，即从入侵检测系统的信息源收集信息，包括系统、网络、数据以及用户活动的状态和行为等。而且需要在计算机网络系统的若干不同关键点（不同网段和不同主机）收集信息，这样可以尽可能地扩大检测范围，而且从一个信息源收集到的信息可能看不出疑点，但从几个信息源收集到的信息的不一致性却是可疑行为或入侵的最好标识。

当然，入侵检测很大程度上依赖于所收集信息的可靠性和正确性，因此，很有必要利用所知道的精确的软件来报告这些信息。因为黑客经常替换软件以搞混和移走这些信息，例如替换被程序调用的子程序、库和其他工具。黑客对系统的修改可能使系统功能失常并看起来跟正常时一样。例如，UNIX 系统的 PS 指令可以被替换为一个不显示入侵过程的指令，或者是编辑器被替换成一个读取不同于指定文件的文件（黑客隐藏了初始文件并用另一版本代替）。这需要保证用来检测网络系统的软件的完整性，特别是入侵检测系统，应具有相当强的坚固性，防止被篡改而收集到错误的信息。

3.1.2　信息分析

入侵检测系统从信息源中收集到的有关系统、网络、数据及用户活动的状态和行为等信息，其信息量是非常庞大的，且绝大部分都是正常信息，只有很少一部分信息才可能表征着入侵行为的发生，要想从大量的信息中找到表征入侵行为的异常信息就需要对这些信息进行分析。可见，信息分析是入侵检测过程的核心环节，没有信息分析功能，入侵检测也就无从谈起。

入侵检测的信息分析方法很多，如模式匹配、统计分析、完整性分析等。每种方法都有其各自的优缺点，也都有其各自的应用对象和范围。

3.1.3 告警与响应

当入侵检测系统检测到攻击或事件以后，系统根据攻击或事件的类型或性质，做出相应的告警与响应，即通知管理员系统正在遭受不良行为的入侵，或者采取一定的措施阻止入侵行为的继续。常见的告警与响应方式包括：

- 自动终止攻击。
- 终止用户连接。
- 禁止用户帐号。
- 重新配置防火墙阻塞攻击的源地址。
- 向管理控制台发出警告指出事件的发生。
- 向网络管理平台发出 SNMP trap。
- 记录事件的日志，包括日期、时间、源地址、目的地址、描述以及事件相关的原始数据。
- 向安全管理人员发出提示性的电子邮件。
- 执行一个用户自定义程序。

3.2 入侵分析的概念

入侵检测技术的研究涉及到计算机、数据库、通信、网络等多方面的知识，一个有效的入侵检测系统不仅能够正确地识别系统中的入侵行为，而且还要考虑到系统本身的安全以及如何适应网络环境发展的需要。所有这些都表明，入侵检测系统将是一个复杂的数据处理系统，所涉及到的问题域中的各种关系也比较复杂。

数据源提供了受保护系统的运行状态和活动记录，而审计数据的处理分析是指对原始数据的同步、整理、组织、分类以及各种类型的细致分析，从而提取其中所包含的系统活动特征或模式，用于对异常和正常行为做出判断。

3.2.1 入侵分析的定义

从入侵检测的角度来说，分析是指对用户和系统活动数据进行有效的组织、整理及特征提取，以鉴别出感兴趣的行为。这种行为的鉴别可以实时进行，也可以事后分析，在很多情况下，事后的进一步分析是为了寻找行为的责任人。

3.2.2 入侵分析的目的

安全管理员希望通过引入入侵行为分析来提高信息系统的安全性。除了检测入侵行为之外，入侵分析还有以下目标：

(1) 重要的威慑力

目标系统使用 IDS 进行入侵分析，对于入侵者具有很大的威慑力，其攻击行为可能会被发现或被追踪。

(2) 安全规划和管理

分析过程中可能会发现在系统安全规划和管理中存在的漏洞，安全管理员可以根据分析

结果对系统进行重新配置，避免被攻击者窃取信息或破坏系统。

(3) 入侵证据

入侵分析可以提供有关入侵行为详细的、可信的证据，用于事后追究入侵者的责任。

3.2.3　入侵分析需要考虑的因素

上节列出了分析处理的目标，本节考虑每一个目标如何驱使入侵检测系统的功能需求，即进行入侵分析时应该考虑的因素，入侵分析需要考虑的因素主要有以下四个方面。

(1) 需求

入侵检测系统支持两个基本需求。一个是可说明性，它是指连接一个活动与人或负责该活动的实体的能力。可说明性要求能够一致地、可靠地识别和鉴别系统中的每一个用户，也必须能够可靠地联系用户及其活动的审计记录或其他事件记录。

尽管可说明性的概念在商业环境中是很普通的，但它们在网络中的实现却相当困难，在网络中一个用户在不同的系统中可能会有不同的身份，就用户本地身份而言，主机级审计跟踪反映了用户的活动，但是在网络中，跟踪用户活动中的身份需要进行额外的处理。

第二个需求是实时检测和响应，包括快速识别与攻击相关的事件链，然后阻断攻击或隐藏系统，避免攻击者的影响。例如，通过跟踪攻击者发出的命令，可以将任何被更改的文件或目标恢复到攻击前的状态。

(2) 子目标

分析也有子目标，例如，用户可能需要在表格中保留信息，用于支持系统和网络上的法庭分析。还可能保留一些系统执行的情况或识别影响系统性能的问题，还包括归档和保护事件日志的完整性等。

(3) 按优先权划分

在目标和要求被计算之后，它们按照优先顺序区分开来。从而决定子系统的结构。优先权可以按进度表划分，也可以按系统划分(例如系统 X 相关的所有需求比其他系统的需求的优先权高)，还可以按照其他属性来划分。

(4) 平衡

有时系统的需求可能和目标有冲突。例如，一个分析目标可能是将分析对目标系统的性能和资源消耗的影响降到最小。然而，为了法律的需求可能需要保存日志，这两个目标就相互冲突，因此需要进行适当的平衡。

3.3　入侵分析的模型

入侵分析是入侵检测的核心功能，从几个观点来看一下入侵检测系统的处理。在这里，定义一个包含能在系统事件日志中找到入侵证据的所有方法的模型。把入侵分析处理过程分为三个阶段：构建分析器、对现场数据进行分析、反馈和提炼。前两个阶段都有三个功能：数据处理、数据分类(即将数据区分为入侵指示、非入侵指示或不确定)、后处理。

3.3.1　构建分析器

在分析模型中，第一阶段的任务就是构建分析器。分析器执行预处理、分类和后处理的核

心功能。不考虑分析方法,如果分析器能够正常运作,它必须能够配合其操作环境,因此即使在独立作为系统环境的一部分被执行的基本系统中,这个阶段也是必须的。

(1) 收集并生成事件信息

构造分析器的第一步是收集事件信息。这个阶段可能收集一个系统产生的事件信息,也可能收集实验室环境下的事件信息,这依赖于分析方法。在有些情况下,根据一套正式的规范工作的开发人员可能会人工收集这些事件信息。

1) 误用检测。对于误用检测,这部分处理包括收集入侵信息,其中有脆弱性、攻击和威胁、具体攻击工具和观察到的重要细节信息。在这种情况下,误用检测也收集典型的一致策略、过程和活动的信息。

2) 异常检测。对于异常检测,事件信息来自于系统本身或指定的相似系统。因此信息是建立指示正常用户行为的基准特征轮廓所必须的。

(2) 预处理信息

所收集的事件信息需要经过许多转换以备分析器使用。它们可能被修改成通用或规范的格式,这种格式通常作为分析器的一部分。在一些系统中,数据也可能被结构化,以便执行一些特性选择或执行其他一些处理。

1) 误用检测。在误用检测中,数据预处理包括转换收集在某种常用表格中的事件信息。例如,攻击症状和策略冲突可能被转换成基于状态转换的信号或某种产品系统规则。在一个基于网络的入侵检测系统中,数据包可能会首先被缓存,并在 TCP 会话期重建。

2) 异常检测。在异常检测中,事件数据可能被转换成数据表,其中一些种类的数据转换成数值表,例如系统名转换成 IP 地址。同样,不同的信息也可能会被转换成一些规范的表格。

规范表格开始用于单个分析器监视多个操作系统。每个操作系统都有其事件数据的本地格式。于是入侵检测开发者们开发了一个可分析来自不同操作系统数据的通用分析器。由此,开发者可集中将新操作系统的事件数据转换为规范格式。规范格式同样也适用于在异种操作系统环境下进行一般分析。有些入侵检测专家认为,许多企业已完全集中于一般常用的操作系统,以至于不再使用规范格式。

(3) 建立一个行为分析引擎

按设计规则建立一个数据区分器,该区分器应该能够把入侵指示数据和非入侵指示数据区分开来。该分析引擎的建立依赖于分析方法。

1) 误用检测。在误用检测中,数据区分引擎是建立在规则或其他模式描述器所描绘的行为上的。这些规则或描述器能分成单一特征或复合特征。例如检查到一个坏格式的 IP 包就属于单一特征,而检测到一个 UNIX 下发送 E-mail 的攻击就属于复合特征。

误用检查区分引擎的结构可以是一个专家系统。专家系统由一个知识库构成,这种知识库包括基于过去入侵可疑行为(例如,处理前阶段收集到的资料)的规则,这些规则通常采用 if-then-else 的结构。

误用检查区分引擎的结构也可以是模式匹配引擎,它把入侵作为攻击特征去匹配审计数据。由于建立在这个模型上的许多系统是可靠而有效的,因此,目前商业入侵检测产品大都采用这种方法。

2) 异常检测。在异常检测中,区分模型通常由用户过去行为的统计特征轮廓构成。这些

特征轮廓也用于标识系统处理的行为，这些统计特征轮廓按照各种算法进行计算，在用户行为模式下其使用方案可能会逐渐变化。这个特征轮廓可以按照固定或可变的进度表进行修补。

(4) 将事件数据输入引擎中

在行为分析引擎建好后，就需要将预处理过的事件数据输入到引擎中。

1) 误用检测。用预处理事件数据或攻击知识的内容，将收集到的对分析引擎有丰富意义的攻击数据输入到误用检测器中。

2) 异常检测。将收集到的参考事件数据输入异常检测器中，系统基于这些数据计算用户轮廓。由于输入到异常检测器的历史数据对入侵来说是不够的，通常假定没有任何协作证据，因此为异常检测器寻找合适的参考事件数据是非常重要的。

(5) 在知识库中保存已输入数据的模型

输入数据的模型被存储到预定的位置，准备操作使用，在这种意义上，输入数据的模型包含了所有的分析标准，事实上也包含了分析引擎的实际核心。

3.3.2　对现场数据进行分析

在对实际现场数据进行分析的阶段，分析器需要分析现场实际数据，识别入侵和其他重要活动。

(1) 输入事件记录

执行分析的第一步是收集信息源产生的事件记录，信息源可能是网络数据包、操作系统审计记录或应用日志文件，并且这些信息源都必须是可靠的。

(2) 预处理

与构造分析器一样，可能需要一些事件数据的预处理。预处理的确切性质依靠分析的性质。例如从高级会话中抽出各种 TCP 消息，并且把来自操作系统的过程标识符构造成一棵高度集成的处理树。

1) 误用检测，对于误用检测，事件数据通常都转换成典型的表格，表格相应于攻击信号的结构。在一些方法中，事件数据被集成，例如用户会话期、网络连接或其他高级事件构成一些重要的微时间片段。在其他方法中，可能会通过捆绑一些属性、完全删除其他属性和在其他数据上进行计算生成新的、条理紧凑的数据记录来精简数据。

2) 异常检测，在异常检测中，事件数据通常精简成一个轮廓向量，行为属性表示为标识。

(3) 比较事件记录和知识库

比较格式化的事件记录和知识库的内容。下一步处理取决于比较的结果和对分析方案的质疑。如果记录指示一次入侵，那么就可以记入日志。如果记录没有指示，分析器就简单地接受下一个记录。

1) 误用检测，在误用检测中，预处理事件记录被提交给一个模式匹配引擎。如果模式匹配器在攻击信号和事件记录中找到一个匹配，则返回一个警告。在一些误用检测器中，如果找到一个部分匹配，如匹配一个指示可能攻击的模式，这种情况可能被记录或缓存在内存中，等待进一步的信息以便作出更明确的决定。

2) 异常检测，在异常检测中，比较用户会话行为轮廓内容与其历史轮廓，依靠分析方案进行判定。如果用户行为与历史行为是足够相关的，则指示不是一次攻击。如果判断用户行为

是异常的,就返回一个警告。许多基于异常检测的入侵检查引擎可能也同时执行误用检测,所以在这些不同的方案中,他们可能相互组合在一起。

(4) 产生响应

如果事件记录是相应于入侵或其他重要行为的,则需要返回一个响应。响应的性质依靠具体分析方法的性质。响应可以是一个警报、日志条目,或者是被入侵检测系统管理员指定的一些其他行为。

3.3.3 反馈和提炼

反馈和提炼是一个非常重要的过程。

1) 误用检测,在误用检测系统中,攻击信息的特征数据库是否可以更新是反映这个阶段的主要功能。每天根据新攻击方式更新攻击信息特征数据库是非常重要的。许多优化的信号引擎能够在系统监控事件数据,不中断分析过程的同时,由系统操作员来更新信号数据库。

大多数基于误用检测的分析方案都有一些关于最大时间间隔的主张,以便在这段时间内匹配一次攻击事件,这段间隔就是著名的事件地平线。对一些系统,事件地平线可能是从用户注册到退出(一次会话)。

在每一个事件中,因为要保存状态信息需要一个大容量的内存(尤其是在比较忙的系统上,有多个用户、进程和网络连接时),状态信息的积极管理是系统稳定性的关键。在基于网络的入侵检测系统(系统在堆上进行 TCP 会话重组)中能够看到这方面的影响。

2) 异常检测,依靠执行异常检测的类型,历史统计特征轮廓被定时更新。例如,在第 1 个入侵检测系统 IDES 中,每天都进行特征轮廓的更新。每个用户的摘要资料被加入知识库中,并且删除最老的资料(如 30 天前的资料)。

对于其余的统计资料(如从第 1 天到第 29 天的资料),将它们乘以一个老化因子。通过这种方法,最近的行为能够更有效地影响正常活动的决策。

3.4 入侵分析方法

入侵检测的分析方法主要由误用检测和异常检测组成。本节将对这两类分析方法中的具体方法进行介绍。

3.4.1 误用检测

误用检测对于系统事件提出的问题是:这个活动是恶意的吗? 误用检测涉及到对入侵指示器已知的具体行为的解码信息,然后为这些指示器过滤事件数据。

要想执行误用检测,需要有一个对误用行为构成的良好理解,有一个可靠的用户活动记录,有一个可靠的分析活动事件的方法。

误用检测最适用于已知使用模式的可靠检测,但它仅能检测到已知的入侵方式。

1. 产品/专家系统

早期执行误用检测的方案之一是使用产品/专家系统。在诸如 MIDAS、IDES、下一代

IDES(NIDES)、DIDS 和 CMDS 中都使用了这种方法。在 MIDAS、IDES 和 NIDES 中，应用的产品系统是 P-BEST，该产品由 Alan Whithurst 设计。而 DIDS 和 CMDS，使用的是由美国国家航空和宇航局开发的 CLIPS 系统。

使用产品系统可以把系统的控制推理从问题的描述中分离出去。这个特性允许用户信息像 if-then 规则一样输入攻击信息，然后输出事实(以审计事件的形式)。系统根据输入的信息评估这些事实。这个过程不需要用户理解产品系统的内部功能。在产品系统之前，用户必须在客户端编写决定引擎和规则的代码，这是一个困难、耗时的任务。

输入的攻击信息使用 if-then 语法。指示入侵的条件被放在规则的左边(if 端)。当满足这些规则时，规则执行右边(then 端)的动作。

与入侵检测产品系统的使用相关的一些实际问题如下：

- 不适于处理大批量的数据。由于产品系统中使用的说明性表达一般用来解释系统实现，解释器总是比编译器慢。
- 不能对连续有序的数据作任何处理。
- 不能处理不确定性。

考虑到在知识系统中由于其他规则的变化影响而必须改变规则时，维护规则系统可能是很有问题的。

2. 状态转换方法

执行误用检测的状态转换方法允许使用最优模式匹配技巧来结构化误用检测问题。它们的速度和灵活性使其具有强有力的入侵检测能力。

状态转换方法使用系统状态和状态转换表达式来描述和检测已知入侵。实现入侵检测状态转换有很多种方法。其中三个主要方法是应用程序编程接口(API)、状态转换的特征、有色 Petri 网和状态转换分析。本节列出用来标识误用模式的策略，然后根据它们去过滤事件数据。

(1) 状态转换分析

状态转换分析是一种方法，它使用高级状态转换图表来体现和检测已知的入侵攻击方式。

这种方法首先在 STAT 系统中进行研究，并且扩展到 UNIX 网络环境 USTAT 中。这两个系统都是由 California 大学开发的。

状态转换图表是贯穿模型的图形化表示。如图 3-1 显示了一个状态转换图表的组成以及如何使用它们来代表一个序列。节点代表状态，弧代表转换。在状态转换表格中，表达入侵的基本思想是：所有入侵者都是从拥有有限的权限出发，并且利用系统脆弱性来获取一些成果。开始点的有限特权和成功的入侵都能作为系统状态来表达。

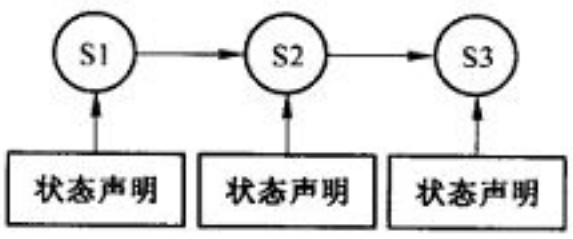

图 3-1　状态转换图表

在使用状态转换图表来表示入侵序列时，系统自身仅限于表示导致一次状态改变的关键活动。初始和入侵状态之间的路径可能是相当主观的，两个人能拿出两个完全不同的代表同样攻击概要的状态转换图表。每一个状态由一个或多个状态声明组成(也表示在图 3-1 中)。

状态转换分析系统利用有限状态机图表模拟入侵。入侵由从系统初始状态到入侵状态的一系列动作组成，初始状态代表入侵执行前的状态，入侵状态代表入侵完成时的状态。系统状态根

据系统属性进行描述,转换由一个用户动作驱动。状态转换引擎保存着一套状态转换图表。在一个给定时间内,假定一系列动作驱动系统到图表中某个特定的状态。当一个新的动作发生时,引擎拿它与每一个图表对比,看是否能驱动到下一个状态。如果这个动作驱动到结束状态,指示一次入侵,则以前的转换信息被送到决定引擎,它向安全人员发出入侵存在的警报。

STAT 方法的优点如下:

- 状态转换图表提供一个直接的、高级的、与审计记录独立的概要描述。
- 转换允许一个人去描绘构成攻击概要的部分顺序信号动作。
- 当渗透成功时,状态转换必须使用最小可能的信号动作子集。因此,检测器能归纳出相同的渗透。
- 系统保存的硬连接信息使它更容易表示渗透情景。
- 系统能检测出协同的缓慢攻击。

STAT 方法的缺点如下:

- 状态声明和信号动作的列表是手工编码的。
- 状态声明和信号可能不能充分表达更复杂的渗透情景。
- 某个状态评估可能要求推论引擎从目标系统获取额外信息。这个处理会导致性能下降。
- 系统不能检测出许多常见攻击,因此必须与其他检测器协作使用。
- 基于该方法的原型系统与其他基于状态转换方法的系统相比效率较低。

(2) 有色 Petri 网和 IDIOT

优化误用检测的另一个基于状态转换的方法是有色 Petri 网方法,由 Purdue 大学研制。这个方法在 IDIOT 系统中实现。

IDIOT 使用一种 CP-Net 的变种来表示和检测入侵模式。在这种模式下,一个入侵被表示为一个 CP-Net。CP-Net 中通过令牌颜色服务来模拟事件上下文。通过审计记录驱动信号匹配,并通过从起始状态到结束状态逐步移动令牌来指示一个入侵或攻击,并且当模式匹配时动作被执行。

这种方法与 STAT 的状态转换方法有显著的不同。首先,在 STAT 中入侵通过作用在系统状态上的效果(也就是入侵的结果)进行检测。在 IDIOT 中,入侵是通过模式匹配构成渗透的特征来进行检测;在 STAT 中是在状态中放置保护,而在 IDIOT 中保护合并在转换处理中。

在 IDIOT 中每一个入侵信号被表示为代表事件和它们上下文关系的一种模式。这种关系模式精确地代表了一次成功的入侵及其企图。CP-Net 图的顶点代表系统状态。入侵模式包括两部分,前面的部分是条件,后面的部分是相关的动作。

这个模式匹配模型由下面几部分组成:

- 上下文描述,允许匹配相关的构成入侵信号的各种事件。
- 语义学,容纳了几种混杂在同一事件流中的入侵模式的可能性。
- 动作规格,当模式匹配时,提供某种动作的执行。

如图 3-2 显示了一个 TCP/IP 连接的 CP-Net 模式。用此方法进行误用检测有许多优点:

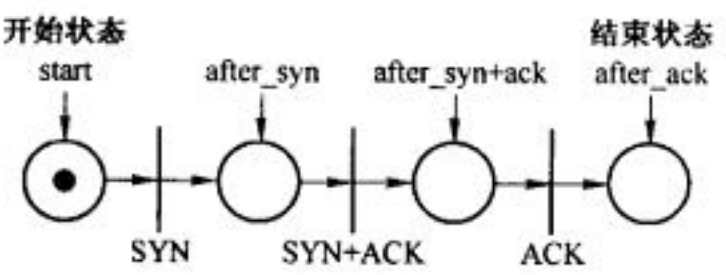

图 3-2 TCP/IP 连接的 CP-Net 模式

- 它是非常快的。在一个非优化 IDIOT 的实验中，每小时激烈活动(产生 C2 审计记录)中匹配 100 个入侵模式，检测器需要 135 秒。与 Sun SPARC 平台每小时产生大约 6 兆审计数据相比，这个结果表明其处理负荷少于 5%。
- 模式匹配引擎是独立于审计格式的，所以它能应用在 IP 包和其他检测问题中。
- 特征在跨越审计记录方面是方便的，因此它们能在不同系统中移动。
- 模式能根据需要进行匹配。
- 事件的顺序和其他排序约束条件可以直接体现出来。
- 系统能提供一个进行成功模式匹配的非常详细的规格，也就是说，它能说明检测到入侵的原因。
- IDIOT 提供一个前端的语言，以便对网络图形描述进行编码。
- 系统允许指定信号匹配时具体执行的动作，因此能够支持自动响应。

IDIOT 具有所有误用检测系统都有的缺陷，即它不能检测出未知的入侵攻击方式。

(3) 基于 API 的方法

优化商业误用检测工具的一般策略是设计一种表格的形式来描述入侵方法。这种表格能够被检测引擎所使用。与前面讨论的一样，尽管已经有一些专家系统语言(如 P-BEST 和 CLIPS)，但它们不是专为入侵检测系统而设计的。有三种方法可以表达入侵以进行误用检测：RUSSEL 语言，STALKER 系统和 N 包过滤语言。

RUSSEL 是一个基于规则的语言，其主要设计目的是用来优化结构数据流的处理，尤其用于操作系统审计记录。RUSSEL 的目标使用户能跨越多主机关联事件并且支持对事件的多级抽象。在一个对异种网络环境系统优化的误用检测系统 ASAX 中使用了 RUSSEL。ASAX 的特点是使用审计数据格式，并提供一个支持可适应规则的组件。

在 RUSSEL 中把事件行为表示为如下形式：条件——→动作。动作能具体到一个允许入侵检测用户对给定检测概要指定响应级别。

用于误用检测目的的另一个描述入侵的方法是使用 STALKER 中使用的方法，STALKER 是一个商用误用检测系统。这个方法使用审计数据格式和一个用于攻击特征的基于状态的数据结构。

这个检测器是通过将审计记录传递到误用引擎来运作的。引擎在状态转换表格中维护了一个检测特征集。特征表达式由包含起始状态、结束状态和每个误用的一个或多个转换功能集的数据结构组成。

这种方法被成功地实现并引入到一系列支持各种操作系统和应用环境的商业产品中去。在 1997 年 NA 公司收购了 Haystack 实验室后，STALKER 产品从市场上退出，但是，这种设计最近已在 Cybercop Monitor 的市场重新出现。

NFR-N 编码提供了一种基于语言的网络监控和分析功能的优化。NFR 是一个网络监控系统，同时也是一些基于网络的商用入侵检测产品的基础。N 编程语言是以字节编码指令操作的解释语言，在一个简单堆栈机器上实现。

这种语言包括流控制、过程和 64 位整数计数器数据类型。它定制成支持网络包过滤器的结构。这些包过滤器能识别网络攻击以及其他网络活动。

过滤器以 N 编码的方式进行编写。N 编码能输入网络监控引擎，进行编译并存储字编码

指令从而优化过滤器性能。使用一个保存每个当前网络会话状态的表格结构来重组网络包传输。状态保存允许在一个连接或传输流的整个生命期中匹配信号。

潜在的引擎也维护关于网络性能的统计资料,包括关于包到达率和网络错误的实时统计资料。引擎也使用一个统计计算来决定在抛弃一个连接状态信息之前保留多长时间。

在 N 编码包过滤机制中,语言把一个网络包接收和一个过滤器捆绑在一起。它也支持其他具体事件。过滤器从包中收集信息,通过警告或记录机制导出这些信息。警告机制向一个警告管理系统发送警告消息,而记录机制向一个用于各种类型后端处理的记录器函数发送一个数据结构。N 语言包含在产品中,可以从开发商处获得其源码。

来自 Cambridge 大学的 Ross Anderson 和 Ahattak 提出了在发现新攻击的入侵检测系统和在攻击被证实后,允许安全管理者找到攻击证据的系统间进行一个功能的分离。为处理后者的问题,Anderson 和 Khattak 提出使用信息检索(IR)技术。这些技术当前广泛应用于 WWW 上的搜索引擎中(例如 Alta Vista)。

IR 系统使用反向文件作为索引,允许高效地搜寻关键字或关键字组合。这些系统也使用一些特定的算法,并使用 Bayesian 推论去帮助提炼搜索。由于这些系统依靠索引,而不是机器学习去发现数据模式,因而有别于数据挖掘。

在成为事实之后,这种方法限制浏览审计信息。研究者构造一个使用 UNIX lastcomm 系统日志和一个由 Arizona 大学开发的搜索引擎 GLIMPSE 的原型。他们使用一个脚本把文件排序成一个比较小的文件集,一个用户一个文件集。进入一个 GLIMPSE 搜索,就可以寻找与特定攻击相关序列相匹配的命令行序列。这种处理可以快速、可靠地定位攻击。

这种方法是简单而有力的,并且使用 IR 技术执行检测活动有比较显著的效率。GLIMPSE 使用的索引是紧凑的(约占索引源的 2%～4%),并且可以作为一个有效的数据精简机制而用于审计跟踪。索引的存在也能作为一个安全机制,揭示黑客改变审计信息来隐藏他们的踪迹。由于 GLIMPSE 和建议的审计数据源是免费的或者包括标准系统包,且这种方法也不昂贵,对许多不能支付其他解决方案的组织来说是一个不错的选择。

3.4.2 异常检测

异常检测需要建立正常用户行为特征轮廓,然后将实际用户行为和这些轮廓相比较,并标识正常的偏离。异常检测的基础是异常行为模式系统误用。轮廓定义成度量集。度量衡量用户特定方面的行为。每一个度量与一个阈值相联系。

异常检测依靠一个假定:用户表现为可预测的、一致的系统使用模式。这个方法也能适应随着时间的流逝在用户行为方面的变化。异常检测的完成仍必需验证(没有人知道任何给定的度量集是否足够完备,并能表示所有的异常行为)。因此,异常检测能否检测出所有感兴趣的情况,并体现出一种对系统的强壮的保护机制,仍需要另外研究。

1. Denning 的原始模型

Dorty Denning 在她 1986 年的论文中,列出了入侵检测的 IDES 模型。她主张在一个系统中可包括 4 个统计模型。每一个模型适合于一个特定类型的系统度量。

(1) 可操作模型

这个模型应用于度量,例如在一个特定时间间隔密码失败次数的事件记数器。这个模型

把度量和一个阈值比较，当度量超出阈值时触发一个异常。这个模型除了应用在异常检测外同样也适用于误用检测。

(2) 平均和标准偏差模型

Denning 的第 2 个检测模型提出典型的数据平均和标准偏差描述。假定所有的分析器已知系统行为度量是平均和标准偏差，这一点由前两个时刻决定。一个新的行为观察如果落在信任间隔之外将被定义为异常。信任间隔定义为一些参数的平均值的标准偏差。Denning 假定这种描述应用到事件记数器、间隔计时器和资源度量。她也提及给这些计算分配权值的能力，以至于最近的数据被赋一个较大的权值。

(3) 多变量模型

Denning 检测模型的第 3 个是多变量模型。多变量模型是对平均和标准偏差模型的一个扩展，是基于两个或多个度量来执行的。因此，可以基于这个度量和相关的另一个度量进行异常检测，而不是严格基于一个度量。所以，不用单独基于一次会话的观察长度来检测一个异常，而是基于这次会话长度和使用的 CPU 周期数之间的关系进行检测。

(4) Markov 处理模型

Denning 模型最后一个，也是最复杂的部分是事件记数器。在这个模型中，检测器把审计事件的每个不同类型作为一个状态变量，并且使用一个状态转换矩阵来描述在不同状态间的转换频率。如果一个新观察事件的或然率太低，这一点是由状态转换矩阵中以前的状态和值决定的，则被定义为异常。这就允许检测器识别不寻常的命令和事件序列，而不仅是单一事件。这时引入了执行事件流的状态分析概念。

2. 量化分析

最常用的异常检测方法是量化分析，其中，检测规则和属性以数值形式表示。Denning 在她的操作模型中涉及这种度量。这套技术经常假定一些计算，包括从简单的加法到比较复杂的密码学计算。这些技术的结果是误用检测信号和异常检测统计模型的基础。这部分描述了几个通常的量化分析并提供一个使用这些技术完成数据精简和入侵检测目标的可操作系统的例子。

(1) 阈值检测

可能最常见的量化分析形式是阈值检测，也称为触发器检测。在阈值检测中，用户和系统行为根据某种属性计数进行描述，这些计数是有某种许可级别的。一个阈值的典型例子是一个系统允许有限的不成功注册次数。实际上每个早期的入侵检测系统包含一个检测规则，根据这种度量定义一个入侵。

其他阈值包括一种特定类型的网络连接数、企图访问文件次数、访问文件或目录次数和访问网络系统次数。在阈值检测中一个固有的假定是在一个特定时间间隔进行度量。这个间隔在时间上可以是固定的(例如，阈值在每天的特定时间重置为零)，或在一个滑动窗口上运行(例如，每过 8 小时进行度量)。

(2) 启发式阈值检测

启发式阈值检测建立在简单阈值检测的基础上，它适合于观察层次。这个处理增加了检测的准确性，尤其当在一个非常宽的用户范围或目标环境中执行检测的时候。例如，可以采用有异常的失败注册时才触发一个警告的异常检测规则，而不是在 8 小时期间失败注册数超过

3 次时就触发一个警告的阈值检测规则。“异常”能通过各种公式定义,使人立即想到的是高斯函数(例如 chi-square),先计算失败注册的平均数,随后将失败注册次数与附加一些标准偏差的平均值进行比较。

(3) 基于目标的集成检查

另一个有价值的量化分析度量是基于目标的集成检查。这是对在一个系统客体中一次变化的检查,这个系统客体通常是不应发生不可预测的变化。对于一个这样的集成检测,最常用的例子是使用一个消息函数计算可疑系统客体的加密校验和。在校验和被计算出来后将其保存在一个安全的地方(如只读介质),系统定期地重新计算校验和,并和储存的参考值进行比较。如果发现了不同,就发出一个警告。

(4) 量化分析和数据精简

在早期入侵检测系统中,量化分析最有意义的一个用途是使用量化度量执行数据精简。数据精简是从庞大的事件信息中删除过剩或冗余信息的处理。这减少了系统存储负荷并优化了基于事件信息的处理。

MADIR 系统是一个使用量化方法支持有效数据精简的例子,该系统由 Los Alamos 国家实验室的计算和通信分公司开发。NADIR 使用数据特征轮廓,把用户活动从审计日志转化成量化的度量向量(它们大部分是线性类型或线性类型和顺序数据的结合)。特征轮廓在时间(每周的总结)和系统(每个系统用户集合的视图)上集成。精简后的数据易于统计,也易于用专家系统来进行检测。

3. 统计度量

第 1 个成功的异常检测系统的例子是基于统计度量的。这些方法包括 IDES、NIDES、还有 Haystack 系统中使用的方法。

(1) IDES/NIDES

IDES 和 NIDES 由 SRI International 公司的研究者开发,是早期最突出的两个入侵检测研究系统。它们都是混合系统,包含误用和异常检测特性,然而这里主要关注统计分析。

在 IDES 和 NIDES 中应用的统计分析技术支持为每个用户和系统主体建立和维护历史统计特征轮廓。这些特征轮廓被定期更新,较老的数据被老化以便于特征轮廓适合反映用户行为在时间上的变化。

系统维护一个由特征轮廓组成的统计知识库。每一个特征轮廓根据一个度量集或度量表示每个用户正常行为。每天一次,基于当天的用户活动,新的审计数据被加进知识库(通过一个指数退化因子来老化旧向量)。

IDES 是这样计算这些统计结果的,每次产生一个审计记录,产生一个摘要测试统计结果。这个被称作 IDES 分数的统计结果通过下面公式计算:

$$IS=(S1,S2,S3,\cdots,Sn)C^{-1}(S1,S2,S3,\cdots,Sn)t;$$

在公式中,$(S1,S2,S3,\cdots,Sn)C^{-1}$是相关矩阵或向量的逆,$(S1,S2,S3,\cdots,Sn)t$ 是向量的转置。每一个 S 度量行为的某一方面,例如文件访问、使用的终端和使用的 CPU 时间。

(2) Haystack

Haystack 是由美国空军开发的异常检测系统,使用两部分统计异常检测方法。第 1 个度量决定一个用户会话与一个已建立的入侵类型的相似程度。这个度量的计算如下:

1）系统维护一个用户行为度量向量。

2）对每一个入侵类型，系统将每个行为度量同一个权值关联，反映度量与给定入侵类型的相关性。

3）对每一个会话，计算用户行为度量向量并与域向量进行比较。

4）注意超出域设置的这些行为度量。

5）累加超出阈的度量相关的权值。

6）基于对所有以前会话加权入侵分数分布，采用累加和给会话分配一个可疑系数的方法。

第 2 部分互补统计方法检测用户会话活动与正常用户会话特征轮廓之间的偏差。这个方法查找显著偏离该用户正常历史统计特征轮廓的会话统计结果。

(3) 统计分析的力度

统计异常检测分析起初是以伪装成一个合法用户的入侵者为目标。尽管统计分析也可检测采用以前未知脆弱性的入侵者，这种入侵不可能被任何其他方法检测到，但这种说法仍然没有在 IDS 系统的产品使用中得到证实。早期的研究者也假设统计异常检测能揭示有趣的、有时是可疑的，能导致发现安全漏洞的活动。这个说法至少在 NADIR(运行在 Los Alamos 国家实验室)中得到证实，该实验室的开发者曾报告说通过使用 NADIR 获得的一些信息发现了系统和安全处理错误，还发现可以改进 Los Alamos 的系统复杂性的一般管理。统计分析的另一个优点是统计系统不像误用检测系统那样需要经常更新和维护。但它依靠几个因素。必须很好地选择度量，充分精细地区分用户行为，也就是说，用户行为的变化必须在相应的度量上产生一个经常的、显著的变化。如果条件满足，不需要对系统进行附带的更改，统计分析可靠地检测到重要行为的机会将是很大的。

(4) 统计分析的不足

当然，统计分析系统也有显著的不足。首先，大部分设计用于执行批模式审计记录处理，没有执行自动响应以防止遭受损害的能力。由于早期系统的设计是从集中的主框架目标平台上监控审计跟踪，所以这种不足在起初并不是一个问题。尽管以后的系统企图执行审计数据的实时分析，所使用和维护用户特征轮廓知识库的内存和处理负荷造成了系统滞后于审计记录的产生。

此外还影响统计分析描述的事件范围。统计分析的本质排除了考虑事件间顺序关系的能力。在大多数系统中，事件发生的确切顺序没有以一个属性形式提供。换句话说，这些事件的水平线限制在一个事件。由于许多指示攻击的异常依赖于这样的顺序事件关系，这种情况体现了这种方法的严重局限性。

在使用量化方法(Denning 的可操作模型)的情况下，很难选择合适的阈值和范围值。

统计分析系统的错误警报率高，会导致用户忽视或禁用系统。这些错误警报包括两种类型错误：类型 1(误报)和类型 2(漏报)。

4. 非参统计度量

由于早期的统计方法都使用参数方法描述用户和其他系统实体的行为模式，所以它们是相似的。参数方法是指分析方法假定了被分析数据的基本分布。例如，在 IDES MIDAS 的早期版本中，用户使用模式的分布假定为高斯分布或正态分布。

当假定不正确时，通过这些假定研究的问题错误率高。当研究者开始搜集系统使用模式，包

括诸如系统资源使用等属性的信息时，发现分布不是正常的，包括这些度量导致较高的错误率。

Tulane 大学提出一种克服这些问题的方法，就是使用非参技术来执行异常检测。这个方法提供用很少的可预测使用模式容纳用户的能力，并允许分析器考虑不容易由参数方案容纳的系统度量。

这种方法涉及非参数据区分技术，尤其是群集分析。在群集分析中，收集了大量的历史数据（一个样本集）并根据一些评估标准（也称为特性）组织成群。执行预处理后，与一特定事件流（经常映射成一具体用户）相关的特性被转化成向量表示（例如，$Xi = [f1, f2, \ldots, fn]$ 表示一个 n 维状态）。群集算法用来把向量分组成行为类，试图使每个类的成员尽可能紧密，而不同类的成员尽可能分离。

非参统计异常检测的前提是根据用户特性把表示用户活动的数据分成两个明显区别的群：一个指示异常活动，另一个指示正常活动。

各种群集算法均可采用。这些算法包括利用简单距离度量一个客体是否属于一个群，以及比较复杂的概念式度量，即根据一个条件集合对客体记分，并用这个分数来决定它是否属于某一个特定群。不同的群集算法通常服务于不同的数据集和分析目标。

Tulane 的研究者发现用资源利用数字作为评估标准，达到这个目标最好的群集算法是 k 最临近算法。该算法用每个向量最邻近的 k 分组向量。k 在样本集中是一个向量数的函数，而不是一个固定值。

使用该分析技术的实验结果显示哪种方式构成的群可以可靠地对相似系统操作分组（例如编译或编辑文件），并且也能根据用户分组活动模式。

非参方法的优点还包括执行可靠精简事件数据能力（在源事件数据到向量的转化中），文档中记录的精简效果有两个以上数量级的改进。其他优点是与参数统计分析相比，检测速度和准确性上的提高。其缺点是涉及超出资源使用的扩展特性将会降低分析的效率和准确性。

5. 基于规则的方法

另一个异常检测的变体是基于规则的异常检测。这个方法的潜在假定与统计异常检测相关的假定是一样的。主要不同是基于规则的异常检测系统使用规则集来表示和存储使用模式。在本节介绍两个这样的方法：Wisdom 方法和基于时间的引导机（TIM）。

（1）Wisdom and sense

第一个基于规则的异常检测系统是由美国 Los Alamos 国家实验室和美国 Oak Ridge 国家实验室的研究者开发的 Wisdom and sense（W&S）系统。W&S 能在多种系统平台上运行并能在操作系统和应用级描述活动。它提出两种移植规则库的方法：手工输入（反映一个策略陈述）和从历史审计记录中产生。规则是通过执行一个种类检查从历史审计记录中派生出来的，解释根据这些规则找到的模式。规则反映了系统主体和客体过去的行为保存在一个树结构中。在审计记录中具体的数据值被分组成线程类，通过线程类关联操作或规则集合。

一个线程类的例子是“所有记录包含同样的用户文件字段值”。每当一个线程相关的活动发生时，在线程中规则就对数据起作用。当转换处理发生时，它们与匹配线程事件进行比较，决定事件是否匹配活动历史模式或代表一个异常，异常现象就是通过这种方式检测出来的。

（2）TIM

Teng、Chen 和 Lu 在数字设备公司工作时提出了 TIM 系统。TIM 使用一个引导方法动

态产生定义入侵的规则。TIM 和其他异常检测系统不同的是，TIM 是在事件顺序中查找模式，而不是在单个事件中查找模式。正如 Denning 在她的启蒙入侵检测著作中建议的一样，TIM 有效地实现了 Markov 转换或然率模型。

TIM 观察历史事件记录顺序，描述事件特定顺序发生的或然率。其他异常检测系统度量单个事件发生是否体现了与正常活动模式的偏差。TIM 集中针对事件发生的顺序，检查一人事件链是否与基于历史事件顺序观察所预期的情况相一致。

例如，设想事件 E1、E2 和 E3 顺序地列在审计跟踪中。TIM 基于它在过去观察的历史顺序描述发生 E1、E2、E3 的或然率。当 TIM 分析历史事件数据时自动产生关于事件的顺序规则，然后在一个规则库中保存这些规则。由于 TIM 对事件顺序进行分组，规则库需要的空间比基于定位单个事件的系统(例如 W&S)需要的空间显著地减少。

如果一个事件顺序匹配了规则头，而下一个事件不在规则实体的预测事件集中，则被认为是异常的。系统通过从规则库中删除预测性很少的规则来提炼它的分析(如果规则 1 比规则 2 成功预测更多事件，那么规则 1 就比规则 2 更具有预测性)。

TIM 与统计度量相比具有显著优点。TIM 非常适合非用户对用户模式的环境，在这种环境下，每个用户在时间上表现出一致的行为。一个大公司可能会有这种环境，不同的用户负责计帐、管理和编程，不同的职责很少交叉运作。这种方法也很适合威胁是与一些事件相关而不是与系统事件完全实现相关的环境。这种方法没有与行为渐变相关的问题。行为渐变是一个与异常相关的失败策略，攻击者随着时间流逝逐渐改变其行为模式，直到训练系统把入侵行为作为正常行为接受。对行为渐变攻击的抵制是由于把语义体现到了检测规则中。

TIM 方法也有缺点，它遇到所有与基于学习方法相关的问题，在这方面，方法效率依赖于训练数据的质量。在基于学习的系统中，训练数据必须反映系统用户的正常活动。进一步来说，这种方法产生的规则不可能复杂到足够去反映所有可能的正常用户行为模式。尤其在系统运作的初期，这种弱点产生大量的错误。错误率高是由于如果一个事件不能匹配任何规则头(也就是说，系统不能在训练数据集中遇到该事件类型)，事件总是触发一个异常。

这个方法是 DEC 公司 Polycenter 入侵检测产品的基础，也是此后许多异常检测研究的基础。

6. 神经网络

神经网络使用可适应学习技术来描述异常行为。这种非参分析技术运作在历史训练数据集上。历史训练数据集假定是不包含任何指示入侵或其他不希望的用户行为。

神经网络由许多称为单元的简单处理元素组成。这些单元通过使用加权的连接相互作用。一个神经网络知识根据单元和它们权值间连接编码成网络机构。实际的学习过程是通过改变权值和建立/删除连接进行的。

神经网络处理涉及两个阶段。在第 1 个阶段，一个代表用户行为的历史或其他样本数据集被移入网络。在第 2 阶段，网络接收事件数据并与历史行为参数比较，决定相似之处和不同之处。

网络通过改变单元状态，改变连接权值，通过建立/删除一个连接来指示一个事件异常。通过逐步修正，网络也更改其关于构成一个正常事件的内容的定义。

神经网络方法需要通过大量假定来进行异常检测。由于它们不使用一个固定特性集来定义用户行为，特性选择是不相关的。神经网络对度量的预期统计分布没有作提前假定，因此这

种方法与其他非参技术相关的典型统计分析相比保留了一些优点。

在使用神经网络进行入侵检测相关问题中，有一种形成不稳定配置的趋势。在这种配置中，网络由于非明显原因学习某些东西失败。无论怎样，使用神经网络进行入侵检测的主要不足是神经网络不能为它们找到的任何异常提供任何解释。这种情况妨碍了用户获得说明性资料或寻求入侵安全问题根源的能力，这使它很难满足安全管理的需要。尽管一些研究者已提出复合方法来解决这些问题，但发表的数据仍然没有说明神经网络方法的可行性。

3.4.3 可代替的检测方案

最近的一些入侵检测方法既不是误用检测也不属异常检测的范围。这些方案可应用于上述两类检测。它们可以驱动或精简这两种检测形式的先行活动，或以不同于传统的观点影响检测策略方式。

1. 免疫系统方法

在一个创新的、有前途的研究项目中，New Mexico 大学的研究者对计算机安全有一个新的看法。研究者提出的问题是“一个人如何用保护自己的方式装配计算机系统?”在回答这个问题时，他们注意到在生理免疫系统和系统保护机制之间有显著的相似性。

上述两个系统运行正常的关键是执行“自我/非我”决定的能力，也就是说，一个组织的免疫系统决定哪种东西是无害实体(例如组织的本身)，哪种是病菌和其他有害因素。由于免疫系统通过使用氨酸、短蛋白质片段做出判断，研究者决定集中在一些认为与氨酸相似的计算机属性上。假设 UNIX 系统调用序列能满足这些要求。

在决定把系统调用作为一个主要的信息源时、研究者考虑了数据的大量目标，包括数据量、可靠检测误用能力和以一种适合高级模式匹配技术编码的适合度。他们决定集中在短顺序的系统调用上，进一步忽略传递给调用的参数，而仅看它们的临时顺序。

系统首先被用来进行异常检测(它也能执行误用检测)。系统按两个阶段对入侵检测分析处理，第 1 阶段建立一个形成正常行为特征轮廓的知识库。因为这里描述的行为不是以用户为中心的，而是以系统处理为中心的，因此这个特征轮廓与本章讨论的其他特征轮廓有一点不同。与这个特征轮廓的偏差被定义为异常。在检测系统的第 2 阶段，特征轮廓用于监控随后的异常系统行为。

源于调用特权程序的系统调用顺序随着时间的推移被收集。系统特征轮廓由长度为 10 的序列组成。使用 3 个度量描述正常行为的偏离。由于 3 个度量允许跨越几种历史上有问题的 UNIX 程序进行多种异常行为的检测，因此它是很有前景的。研究也显示执行顺序集是十分紧凑的。

随后的研究是对几种描述正常行为的不同方法进行比较。当研究监控更复杂的系统时，是否有更有力的数据模型方法，以显著改进这个方法的性能。有些令人意外的是，甚至有力的数据模型技术(例如，隐 Markov 模型，尽管计算量很大，但却是十分可靠的)，也不能给出比基于时间的较简单的顺序模型明显好的效果。

尽管自我/非我技术是十分有力和有希望的方法，但它不能彻底解决入侵检测问题。一些攻击包括伪装和策略违背，不涉及特权处理的使用。因此，使用这种方法不易检测出这些攻击。

2. 遗传算法

另一个比较复杂的执行异常检测的方法是使用遗传算法执行事件数据分析。

遗传算法是进化算法的一个实例。进化算法吸收达尔文自然选择法则(适者生存)来解决问题。遗传算法用允许染色体的结合或突变以形成新个体的方法来使用已编码表格(也称为染色体)。这些算法在多维优化问题处理方面的能力已经得到认可。在多维最优化问题中,染色体由优化的变量编码值组成。

入侵检测处理包括为事件数据定义假设向量,向量指示一次入侵或指示不是一次入侵。然后测试假设是否是正确的,并基于测试结果尽力设计一个改进的假设。重复这个处理直至找到一个解决方法为止。

在这个处理中遗传算法的作用是设计改进的假设。遗传算法分析包括两步。第 1 步是用一个位串对问题的解决办法进行编码。第 2 步是与一些进化标准比较,找一个最合适的函数测试群体中的每个个体。

由 Supelec 和法国工程大学开发的 GASSATA 系统中,遗传算法使用一个假设向量集,n 维(n 是潜在的已知攻击数)的 H(每个重要事件流对应一个向量)被应用到区分系统事件问题上。如果 Hi 代表一次攻击则定义为 1,否则为 0。

最适合的函数有两部分。首先,将一个特定攻击对系统的危险性乘以假设向量值。然后由一个二次消耗函数对结果调整,删除不实际的假设。这一步改进了在可能攻击间的区别。处理的目标是优化分析的结果,以判断一个已检测攻击是真实的(或然率接近于 1)或一个已检测攻击是错误的(或然率接近于 0)。

遗传算法对异常检测的实验结果是令人满意的。在实验操作中,检测的平均或然率(现实攻击的准确检测)是 0.996,误报的平均或然率(没有攻击的检测)是 0.0044。所需的构造过滤器的时间也是令人满意的。对于一个 200 次攻击的样本集,一般用户持续使用系统超过 30 分钟才能生成的审计记录,该系统只需 10 分 25 秒即可完成。

使用遗传算法进行误用检测有以下缺点:

- 系统不能考虑由事件缺席描述的攻击(例如,"程序员不使用 cc 作为编译器"规则)。
- 由于个别事件流用二进制表达形式,系统不能检测多个同时攻击。
- 如果几个攻击有相同的事件或事件组,并且攻击者使用这个共性进行多个同时攻击,系统不能找到一个优化的假设向量。
- 系统不能在审计跟踪中精确地定位攻击。因此,不会有临时性出现在检测器的结果中(这点不足与神经网络方法中提到的问题相似)。因此,如果要求这种支持,遗传算法方法必须有后备 post hoc 搜寻或其他研究的援助。

3. 基于代理检测

基于代理的入侵检测方法就是一个在主机上执行某种安全监控功能的软件实体。它们自动运行,仅由操作系统而不是其他进程控制。基于代理的方法连续运作,并且除了与其他相似的结构代理交流和协作外,还从经历中学习。

基于代理的检测方法是非常有力的。根据开发者的最初想法,一个代理可以是很简单的(例如记录在一个特定时间间隔内,一个特定命令触发的次数),也可以是很复杂的(例如在一个特定环境中寻找一系列攻击的证据)。

代理的能力范围允许基于代理的入侵检测系统提供一个异常检测和误用检测的混合能力。例如,一个代理可以设计成使它的检测能力适应本地环境变化。它也能在一个很长时间内监控非常不确定的模式,因此能检测缓慢攻击。最后,一个代理能对一个检测到的问题制定非常精细的响应(例如改变一个进程的优先级,有效地使它慢下来)。

(1) 入侵检测的自动代理

一个基于代理的入侵检测系统原型是入侵检测的自动代理(AAFID),由 Purdue 大学的开发者研制。本节以它作为基于代理解决方法讨论的基础。

基于代理的入侵检测系统体系要求代理的一个分层顺序控制和报告结构。一个主机上能驻留任意数量的代理。在一个特定主机上的所有代理向一个单独的入侵检测的自动代理报告它们的发现。收发器也对代理报告的信息执行数据精简并向一个或多个监控器、下一级分层报告结果。

可以是多层分层结构的监控器控制和合并来自多个接收器的信息。由于 AAFID 的体系结构允许冗余接收器汇报信息,一个监控器的失效不会危及入侵检测系统的操作。监控器有能力从整个网络访问数据,然后执行来自接收器结果的融合。这个特点使系统能检测多主机攻击。通过一个用户接口,系统用户输入命令控制监控器。反过来,它们基于这些用户命令控制接收器。

每个组件提供了 APIs 完成代理、接收器、监控器和用户间的通信。

AAFID 和其他基于代理的方法相比优点如下:

- 比其他入侵检测系统对插入和逃避攻击更具有抵御力。
- 需要时,体系结构提供加入新组件或替换旧组件的能力,更容易缩放。
- 在部署代理之前,能独立于整个系统进行测试。
- 因为代理能互相通信,因此能成组部署,每个代理执行不同的简单功能,但却为一个复杂结果服务。

与 AAFID 体系相关的缺点如下:

- 监控器是单独失效点。如果一个监控器停止工作,它控制的所有接收器停止产生有用信息。存在可能的解决策略,但它们仍然没有被测试。
- 如果备份监控器用于解决第 1 个问题,处理信息一致性和备份是很困难的。这种情形要求附加机制。
- 缺少允许不同用户对入侵检测采用不同访问模式的访问控制机制。这个显著缺陷存在于每一级体系中。
- 由于攻击者证据到达监控器有传播时间,这会导致发生问题。这个问题是所有分布入侵检测系统共有的。
- 与入侵检测其他部分一样,在设计入侵检测系统用户接口时需要更多的洞察力。这种洞察力涉及到从演示方案到方针结构和具体方案。

(2) EMERLD

使用分布代理方法进行入侵检测的第 2 个体系是 EMERLD 系统,是由 SRI International 公司研究并开发出的原型系统。EMERLD 在一个框架中包括大量本地监控器,这个框架向一个全局检测器数组支持分布的本地结果,反过来,全局检测器数组确认警报和警告。

像 SRI International 公司以前的入侵检测系统 IDES 和 NIDES 一样，EMERLD 也是一个复合入侵检测器，使用特征分析和统计特征轮廓来检测安全问题。

值得注意的是由于 EMERLD 把分析语义从分析和响应逻辑中分离出来，因此在整个网络上更容易集成。EMERLD 也具有在不同抽象层进行分析的重要能力。进一步来说这种设计支持现代入侵检测系统中的另一个重要议题——协作性。

EMERLD 的中心组件是 EMERLD 服务监控器，它在形式和功能上和 AAFID 自动代理相似。服务监控器可编程执行任何功能并且可以部署在主机上。为支持分层数据精简，服务监控器进行分层，执行一些本地分析并向高级监控器报告结果和附加信息。这个系统也支持大范围的自动响应，对负责大规模网络的用户来说是非常重要的。

由于 EMERLD 是建立在相当数量的组织的洞察力（集中于 IDES 和 NIDES 中）之上的，因此，在保护大的分布式网络方面有很大的优势。

4. 数据挖掘

与一些基于规则的异常检测相似的方法是使用数据挖掘技术建立入侵检测模型。使用这种方法可以发现能用于描述程序和用户行为的系统特性一致使用模式。而系统特性集由引导方法处理形成识别异常和误用的分类器（检测引擎）。

数据挖掘指从大量实体数据中抽出模型的处理。这些模型经常在数据中发出对其他检测方式不是很明显的事实。挖掘审计数据最有用的 3 种方法是：分类、关联分析和序列模式分析。

- 分类给几个预定义种类中的每一个赋一些数据条目。分类算法输出分类器，例如判定树或规则。在入侵检测中，一个优化的分类器可靠地识别落入正常或异常种类的审计数据。
- 关联分析识别数据实体中字段间的自相关和互相关。在入侵检测中，一个优化的关联分析算法识别最能揭示入侵的系统特性集。
- 序列模式分析使序列模式模型化。这些模型能揭示典型的同时发生的审计事件并且拥有扩展入侵检测模型，包括临时统计度量的密钥。这些度量能提供识别拒绝服务攻击的能力。

研究者已开发出标准数据挖掘算法扩展来适应一些审计和其他系统事件日志的特殊需求。使用现场数据实验初始结果是很有效的。

3.5　告警与响应

在完成系统安全状况分析并确定系统问题之后，就要让人们知道这些问题的存在，在某些情况下，还要另外采取行动。在入侵检测处理过程模型中，这个阶段称之为响应期。理想情况下，系统的这一部分应该具有丰富的响应功能特性，并且这些响应特性在针对安全管理小组中的每一位成员进行裁剪后，能够为他们都提供服务。

本节将阐述入侵检测系统处理分析结果的方法，并且概要描述对所检测出的问题作出响应的选择模式。这些选择模式包括：被动响应和主动响应。被动响应就是系统仅仅简单地记录和报告所检测出的问题，而主动响应则是系统要为阻塞或影响进程而采取行动。

3.5.1 对响应的需求

在设计入侵检测系统的响应特性时,需要考虑各方面的因素。某些响应要设计得符合通用的安全管理或事件处理标准,而另一些响应要被设计来反映本地管理的策略。在为商业化产品设计响应特性时,商家应该给最终用户提供这样一种性能:用户能够剪裁定制其响应机制,以使其符合特定的需求环境。

在入侵检测系统研究和设计的早期,绝大多数设计人员把重点放在系统的监视和分析部分,而将响应部件留给用户去设计并嵌入。对于"在响应部件中什么是用户真正需要的问题"有过大量的讨论,但没有人对可能要装载和使用入侵检测系统的操作环境的模式有一个非常清楚的认识。

一个随时都可能出现的问题就是"用户"的含义,也就是说,究竟谁是一个入侵检测系统的"标准用户"?

根据实际情况,可以把入侵检测系统的用户分成 3 类。第 1 类用户是网络安全专家或管理员。安全专家有时仅作为系统管理小组的咨询顾问。这些安全专家要与各种商业性的入侵检测系统打交道,他们非常熟悉各种入侵检测工具。然而这些安全专家并不总是熟悉他们正在监视测试的网络系统。

第 2 类用户是系统管理员,他们使用入侵检测系统来监控和保护他们所管理的系统。在某些情形下,他们是入侵检测系统的强有力的用户,因为他们对检测工具和保护的网络环境都有很好的技术性理解。系统管理员也是对入侵检测系统要求最高的用户,有时候他们所需求的特性很少被其他的产品用户们所使用。

第 3 类用户是安全调查员,他们是系统审计小组或法律执行部门的成员,他们使用入侵检测产品来监视系统运行是否符合法规或者符合某一项调查。这些用户可能不具备技术理论基础来理解入侵检测工具或正在运行的系统。然而,他们对调查一个问题的过程非常熟悉,并且给入侵检测系统设计者提供重要的知识来源。

本节中,我们分别相应地称这 3 类用户为:安全管理员、系统管理员和调查员,而术语"用户"一词则包括所有这 3 类人员。

用户依靠入侵检测系统来对大量的系统事件记录数据进行复杂而准确的分析。最终,他们希望系统可靠而精确地运行,并且在相应的时刻直接将分析的结果以易于理解的术语形式传送给最需要它的有关人员。虽然对用户的需求可能要作各种各样的考虑,但系统目标总是一致的。

1. 操作环境

在设计一种响应机制时,要考虑入侵检测系统运行环境的特性,对于有很多控制台并直接连接于网络运行中心的入侵检测系统,其报警和通知的要求与家庭办工的桌面入侵检测系统有很大的不同。

作为通知的一部分,入侵检测系统所提供的信息形式也依赖其运行环境。网络运行中心的职员可能比较喜欢能提供低层网络流量详细资料(例如分片包的内容)的产品。而一个值夜班的安全管理员可能认为入侵检测系统能在适当的时刻给合适的人员提供一个告警信息就足够了,其他信息不会有什么价值。

当一个人要负责监视多个入侵检测系统时，安装声响告警器非常适合。而这种告警模式对于从单一控制台来管理一个复杂网络的多个操作而言是一件很麻烦的事情。

对于全天守候在系统控制台前的操作员，安装可视化告警和行动图表会更有价值。当监视其他安全设施的部件(如加密系统或防火墙等)在管理区域不可见时，这种可视化告警和行动图表也特别有帮助。

可视化告警对不能在现场观看它们的操作员来说，可能价值不大。彩色编码的告警状态显示对色盲的操作员或视力有障碍的操作员是没有什么意义的。

2. 系统目标和优先权

推动响应需求的另一个因素是所监控的系统功能。对为用户提供关键数据和业务的系统，需要部分地提供主动响应机制，例如，对被确认为攻击源的用户能终止其网络连接。这类系统的一个例子是医院急救室的医疗记录服务器，另一个例子是高流量、高交易收入的电子商务网站的 Web 服务器。在这类情形下，一次成功的拒绝服务攻击会造成灾难性影响。在上述两个例子中，始终保持系统服务的可用性所产生的价值远远超过在入侵检测系统中提供主动响应机制所造成的额外费用。

3. 规则或法令的需求

产生特殊响应性能的另一些因素包括入侵检测方面的规章或法令的需求。在某些军事计算环境里，入侵检测系统需要这样一些性能：能使某些类型的处理过程发生。例如，只有当入侵检测系统在工作时，一个系统才被授权处理一定水平的敏感性的分类信息。在这些环境里，规则控制着入侵检测系统的操作，事件报告需求控制着入侵检测系统运行结果的表达格式和传送时间的安排。如果入侵检测系统不再运行，则规则规定指定秘密级别的信息不能在系统上运行。

在在线股票交易环境中，安全和交易代理要求交易系统在交易期间对客户是可接入的，任何接入的拒绝将使站点遭受罚款或赔偿。这种情形既要有自动响应机制使其能阻塞攻击，使正常客户服务工作良好；又要能对检测出的问题作出简要的解释说明，进而使灾难后的恢复工作尽可能快地完成。

4. 给用户传授专业技术

入侵检测产品往往忽略的一种需求是随同检测响应或作为检测响应的一部分为用户提供指导。也就是说，系统应该将检测结果连同解释说明和建议一起呈现给用户，以使用户采取适当的行动。正是在这方面，入侵检测产品之间产生巨大差异。一套设计良好的响应机制能构筑好信息和解释说明，以指导用户进行一系列的决策，采用合适的命令，最终引导用户正确地解决问题。

这种响应机制也允许针对具有不同专业技术水平的用户对检测结果的表现形式进行剪裁。在对用户的分类描述时也应提到注释，不同的入侵检测系统用户具有不同的信息需求。系统管理员可能能明白网络服务请求序列或原始的数据包的含义，安全专家也许能理解“端口扫描”与“邮件发送缓冲区溢出”两者之间的区别。调查员可能需要这样一种性能：能追踪一个特别用户所操作的命令序列，以及这些操作给系统带来的影响。

入侵检测系统开发者应该能使其产品适应各种不同用户的能力和专业技术水平。在这样一个快速成长的市场里，专家型的用户在若干年以后可能会越来越少。

3.5.2 响应的类型

入侵检测系统的响应可分为主动响应和被动响应两种类型。在主动响应里,入侵检测系统应能阻塞攻击或影响进而改变攻击的进程。在被动攻击里,入侵检测系统仅仅简单地报告和记录所检测出的问题。

主动响应和被动响应并不是相互排斥的。不管使用哪一种响应机制,作为任务的一个重要部分,入侵检测系统应该总能以日志的形式记录下检测结果。

在网络站点安全处理措施中,入侵检测的一个关键部分就是确定使用哪一种入侵检测响应方式以及根据响应结果来决定应该采取哪一种行动。

1. 主动响应

主动响应检测到入侵后立即采取行动。对主动响应,有许多种选项可供选择:

- 对入侵者采取反击行动。
- 修正系统环境。
- 收集尽可能多的信息。

虽然第1个选项,即对入侵者采取反击行动,在一些团体里特别流行,但它不是惟一的主动响应。由于它还涉及到重要法规和现实问题,所以这种响应也不应该成为用户最常用的主动响应。

主动响应有两种形式:一种是由用户驱动的,一种是由系统本身自动执行的。

(1) 入侵者采取反击行动

主动响应中第1个选项是对入侵者采取反击行动。许多信息仓库管理小组的成员都设想这种选项最富进攻性的形式是:追踪入侵者的攻击来源,然后采取行动切断入侵者的机器或网络的连接。那些长期受到安全困惑的安全管理员往往面对很多黑客拒绝服务式攻击,这种方法对他们也是很有吸引力的。

但是,这个选项本身也会引出一个很大的安全纰漏。对攻击者的反击带来的危险性包括如下几个方面:

- 根据网络黑客最常用的攻击方法,被确认为攻击你的系统的源头系统很可能是黑客的另一个牺牲品。黑客先成功地黑掉一个系统,然后使用它作为攻击另一个系统的平台,这种网络接力是普通的作法。如果你瞄准这个攻击源头系统,就很可能在反击一个无辜的同伴。
- 即使攻击者确实来自一个其合法控制的系统,但攻击源的IP地址欺骗也是常有的事。所以,看起来攻击你的系统的源头IP地址实际上可能是从另一个无辜牺牲者"借"来的。
- 有时候,简单地反击会惹起对手更大的攻击。攻击开始时可能只是对你的系统进行常规的监视或扫描,有可能发展成为全范围的敌意攻击,使你的系统资源的可用性处于危险之中。
- 在许多情况下,反击会使你自己冒违法犯罪的风险,如果你的行为攻击了一个无辜的一方,该方可能要控告你,并要求赔偿其损失。更进一步,你的反击本身可能违反了计算机法令法规,可能要吃官司。最后,如果你在政府部门或军事组织工作,你可能在违反

策略，并且会受到纪律处分或被解雇。因此，碰到此类事情，要与权威部门联系，以求得他们的帮助，来对付和处理攻击者。

对入侵者采取反击行动也可以以温和的方式进行。例如，入侵检测系统可以简单地通过重新安排 TCP 连接来终止双方网络会话。系统也可以设置防火墙或路由器阻塞来自看起来像是入侵来源的 IP 地址的数据包。

另一种响应方式是自动地向入侵者可能来自的系统的管理员发 E-mail，并且请求协助确认入侵者并处理相关问题。当黑客通过拨号连接进入系统时，这种响应方式还能产生多种用途。随着整个通信基础设施中跟踪能力的不断增强，该响应可以使用电话系统的特性(例如呼叫者标识或陷阱和跟踪)来协助建立入侵者的档案。

- 基于用户驱动的响应。许多主动响应性能源自超级安全管理者手忙脚乱地执行响应的时代。虽然响应中的性能能实时地自动处理攻击，但并不意味着这是一种可取的方式。

例如，假设攻击者发现你的系统对拒绝服务式攻击的自动响应是“避开”(也就是，终止目前的连接并拒绝以后该源 IP 地址的 TCP 连接)表面的攻击源，则攻击者可以使用 IP 地址欺骗工具来对你的系统进行拒绝服务式攻击，而攻击好像来自用户的一些最重要的客户，从而导致那些客户被拒绝访问你的关键资源。更糟糕的是，严格地来讲是你的入侵检测系统使系统拒绝服务。

- 自动响应。另一方面，使一些基本的主动响应自动化是必要的，因为攻击进行的速度很快。绝大多数来自 Internet 的攻击一般使用攻击软件和脚本。这些攻击以阻止手工干预的步调来进行。入侵检测的设计者们应该考虑单独一个主动响应能否用手工处理。如果干预必须自动进行，应该采取衡量措施以使主动响应机制对付攻击带来的风险最小。

(2) 修正系统环境

主动响应的一个选项是修正系统环境，这通常是一种最佳的响应方案，特别是与提供调查支持的响应结合在一起的时候。修正系统环境以堵住导致入侵发生的漏洞的概念与许多研究者所提出的关键系统耦合的观点是相一致的。“自愈”系统装备着类似于人体免疫系统的防卫设备，该设备能识别出问题所在，能隔离产生问题的因素，并对处理该问题产生一个适当的响应。

在一些入侵检测系统中，这类响应也许通过增加敏感水平来改变分析引擎的操作特征。它也能通过插入改变专家系统，即通过这些规则提高对某些攻击的怀疑水平或增加监视范围，以比通常更好的采样间隔收集信息。这种策略类似于实时过程控制系统反馈机制，即目前系统处理过程的输出将用来调整和优化下一个处理过程。

(3) 收集额外信息

主动响应的第 3 选项是收集额外信息。当被保护的系统非常重要并且系统的主人想进行法则矫正时，这种选项特别有用。有时，这种日志响应是和一个特殊的服务器相结合配套使用的，该服务器用来营造环境使入侵者被转向。这种服务器有许多称呼，最常用的是“蜜罐”(honey pots)、“诱饵”(decoys)和“玻璃鱼缸”(fishbowls)。这些服务器装备着文件系统和其他带有欺骗性的系统属性，这些属性被设计用来模拟关键系统的外在表像和内容。

1992 年，Bill Cheswick 第一次探索“诱饵”服务器的具体步骤，一个攻击 Cheswick 系统的荷兰黑客就是被重定向进入该服务器的。Cliff Stoll 在他的经典著作里介绍过“诱饵”服务器的使用情况。

"诱饵"服务器对那些正收集关于入侵者的威胁信息或收集对入侵者采取法律行动的证据的安全管理者来说是很有价值的。使用"诱饵"服务器使入侵的受害者在实际系统的内容没有毁坏或暴露风险的情况下可以确定入侵者的意图、记录下入侵者入侵行为的额外信息。这些信息也能用来构造用户检测信号。

以这种方式收集的信息对那些从事网络安全威胁趋势分析的人来说也是有价值的。这种信息对那些必须在有敌意威胁的环境里运行或易遭受大量攻击的系统(例如政府 Web 服务器或具有商业价值的电子商务网站)是特别重要的。

2. 被动响应

被动响应就是那些只向用户提供信息而依靠用户去采取下一步行动的响应。在早期的入侵检测系统里,所有的响应都是被动的。然而,被动响应是很重要的,在一些情形下是系统惟一的响应形式。本节根据对危险程度的大小顺序列出各种被动响应,在告警机制和问题报告之间危险程度是完全不一样的。

(1) 告警和通知

绝大多数入侵检测系统提供多种形式的告警生成方式以供选择。这种弹性允许用户裁剪告警使其适合本组织的系统操作程序规范。

- 告警显示屏。入侵检测系统提供的最常用的告警和通知方式是屏幕告警或窗口告警消息。它出现在入侵检测系统控制台上,或出现在入侵检测系统安装时由用户配置的其他系统上。在告警消息方面,不同的系统提供的信息详实程度不同,范围从一个简单的"一个入侵已经发生"到列出此问题的表面源头、攻击的目标、入侵的本质以及攻击是否成功等广泛性记录。在一些系统里告警消息的内容也可以由用户定制。
- 告警和警报的远程通知。按时钟协调运行的系统使用另一种告警形式。在这些情形下,入侵检测系统能通过拨号寻呼或蜂窝电话向系统管理员和安全工作人员发出告警和警报消息。E-mail 消息是另一种通知手段,虽然这种方法在攻击连续不断的情况下是不主张使用的,因为攻击者可能会读取 E-mail 消息,更糟糕的是,他们可能会阻塞 E-mail 消息。在一些情形下,通知选项允许用户给相应单位配置附加信息或告警编码。

(2) SNMP Trap 和插件

一些入侵检测系统被设计成与网络管理工具一起使用。这些系统能使用网络管理基础设施来传送告警,并可在网络管理控制台显示告警和警报信息。一些产品就依附简单网络管理协议(SNMP)的消息或 SNMP Trap 作为一个告警选项。

在一些商业化产品里目前还提供这个选项,但相信入侵检测系统和网络管理系统能够更多更彻底地集成在一起。许多好处与这种集成相关,包括使用常用通信信道的能力,以及在考虑网络环境时对安全问题提供主动响应的能力。进一步,SNMP Trap 允许用户将响应一个检测问题相关的处理负荷转移到接收该 Trap 并对其采取行动的系统中去运行。

3.5.3 调查期间掩盖跟踪

一个入侵检测系统的部分效力依赖于它提供对攻击者隐蔽的、可靠的监视能力。当一个系统正在遭受攻击时,以攻击者不可见的方式处理告警和通知是明智的做法。这种方法允许

入侵者的会话连接仍在进行时从事调查活动，允许建立行为责任说明。在这些情形下，告警和通知可以在加密信道上进行。

1. 对响应部件自动防故障装置的考虑

入侵检测系统的响应部件里应该采取一些自动防故障装置的措施。

首先，和系统其他部件一样，响应系统和该部件所有成分的设计应假设对手将把它们作为攻击目标的一部分。攻击策略可能包括监视响应信道、搜寻检测迹象或者切断或截取告警和警报信道，使操作员得不到有关攻击的通知。

为使系统可靠运行，使用加密信道以及其他密码学手段，如隐藏与认证入侵检测的通信是完全必要的。这种措施能排除多种以入侵检测系统的响应部件和其他部件为目标的攻击。

入侵检测系统生成的告警信息应该是有冗余的，并可以使用多种信道进行传递。例如，用户可能想设置入侵检测系统，使得当攻击者对关键系统进行严重攻击时触发 3 次告警，通过正常网络通信给通知单元传送第 1 个警报，通过加密信道给通知单元传送第 2 个警报，通过拨号信道给通知单元传送第 3 个警报。

应该保护好由响应部件生成的证明所有检测结果的日志式记录以免遭修改或破坏。因为它们在调查时可能要用来支持任何法律行动，所以这种保护特别重要。保护这些日志记录的一种方法是采用只写一次的 CD-ROM 以及一个优化的硬拷贝备份给行式打印机。对特别关键的系统，推荐使用冗余的日志记录机制。

2. 处理错误告警

在入侵检测系统中存在并且需要在系统的响应部件嵌入智能的问题涉及到错误告警。这些错误告警可能是误报，即系统没有攻击发生却识别出攻击。也可能是漏报，即系统发生攻击却不能识别出这些攻击。漏报是一个分析性问题并且在响应部件中是最好处理的。然而，误报说明响应部件本身就有问题。

当一个有缺陷的网络部件破坏信息包时，会立即触发一个错误的网络攻击提示，分析系统就会向响应部件传送告警，响应部件相应地生成告警信息。如果被破坏的信息包每秒只触发两次错误告警，则响应部件的服务请求就会泛滥并且最终导致系统崩溃。因此，响应部件应该给用户提供在一个时间间隔里由同一源头的信息包触发的告警次数。这项技术不仅可用来保护响应部件的完整性，而且可减少用户忽略由检测器生成的告警的可能性。

3. 存档和报告

被动响应的一项长期的重要任务就是将检测结果存档以备日后使用。这些入侵检测系统以数据库的形式存储检测结果。这种方法使用户可以生成各种报告，以满足每一个需求者的需要。在入侵检测产品中该特性非常流行，因为它允许安全管理人员有规律地通报执行管理小组关于系统安全的状态，进而可以把问题的详尽细节转给装备好用来处理该问题的部件。

对安全处理很重要的另一个任务是维护入侵检测结果的日志记录，并以系统日志和审计跟踪完全相同的方式将它们结构化。这个日志记录应该写到一次性的 CD-ROM 中以使它们免遭修改和删除。这个日志记录很重要，因为它提供了长期的连续的对系统的入侵记录。这些材料作为长期问题的进展的记录文件也具有重要性，并且更重要的是能作为组织决定寻求合法修补问题的依据。不管进行的修补是违法的还是合法的，这种依据非常关键。

3.5.4 按策略配置响应

一个成功的安全管理计划要有效地将策略和支持结合在一起。为优化入侵检测系统的使用,应该把组织的安全策略和程序考虑进去。着手此项工作的一种方法是提供详细清单说明哪一种行动对应着哪一种被检测出的入侵或安全破坏。这些行动按其响应发生时间和紧急程度顺序分为下列4类:立即或紧急行动、适时行动、本地的长期行动和全世界的长期行动。

1. 立即或紧急行动

立即或紧急行动要求系统管理员立即跟踪一个入侵或攻击。这些行动包括如下4个方面:

- 初始化事件处理进程。
- 执行损失控制和侵入围堵。
- 通知执法部门或其他组织。
- 恢复受害系统服务。

关于立即或紧急行动的响应时间跨度可能取决于本地的策略,并且能随着攻击的激烈程度进一步改进。

2. 适时行动

适时行动要求系统安全管理跟踪检测到的攻击或安全破坏。距问题出现时刻的时间范围可以从几个小时到几天,并且这些行动通常紧随于立即或紧急行动之后。应该适时发生的行动包括如下7个方面:

- 人工调查不常规的系统使用模式。
- 调查和隔离检测到的问题的根源。
- 若有可能,通过向开发商申请补丁或重新配置系统来改正或纠正这些问题。
- 向合适的权威组织报告事件的详细细节。
- 在入侵检测系统中改变或修正检测信号。
- 通过法律手段对付犯罪者。
- 处理与攻击相关的公共问题,并且把此事通知股东、规则制定者和其他有合法报告需求的人员。

3. 本地的长期行动

本地的长期行动是指那些虽然没有立即或紧急行动和适时行动那么紧急,但对安全管理过程一直是很重要的系统管理行动。这些行动对本组织的影响是局部或本地的。这些行动可能会作为规则调整的一部分来作进度编排。

这类行动包括如下两个方面:

- 编制统计报表和进行趋势分析。
- 追踪发生过的入侵模式。

应该对这些入侵和安全破坏的模式进行评估以确定他们需要修改或改进的程度。例如,许多以已经被修改过的脆弱性为目标的攻击可能会导致的安全策略需求是:系统软件应该定期修补。许多的错误告警可能表明需要重新评估入侵检测系统的检测信号或配置,或者寻找另一个可替代的入侵检测产品。最后,由于用户的错误而导致的大量的问题则表明需要对用户进行额外的培训。

4. 全局的长期行动

全局的长期行动是指那些对整个社会的安全状态虽不关键但也很重要的系统管理行动。这些行动的影响不仅仅局限于本组织，很可能要由一个工业组织或社团协调一致地指导进行。

这类行动包括如下几个方面：

- 要让商家知道由于他们产品中的安全问题而使本组织遭受该问题的困扰。
- 与立法者和政府部门沟通使之对系统安全威胁进行附加的法律修补。
- 向执行部门或其他维护统计资料的组织报告关于安全事故的统计报表。

许多系统安全中的关键问题不可能在本地或局部就能简单地解决。要靠全社会的行动和努力才能较好地参与和解决这些问题。

3.6　入侵追踪

要有效地进行入侵检测，除了技术手段外，还经常需要一定的经验和技巧，因为一些技巧往往会起到事半功倍的效果。在本节中，将讨论网络上反攻击的一些小的经验和技巧，这些经验和技巧在反击黑客的攻击中可以产生比较理想的效果。

在局域网上可用“广播模式”的信息发送方法，此种方法不指定收信端，只要和此网络连结的所有网络设备都是收信对象。但是这仅仅在局域网上能够实行，因为局域网上的机器不多。如果像 Internet 上有成千上万的主机，根本就不可能实施信息广播。因此，任何局域网内的路由器或是类似网络设备都不会将自己局域网内的广播信息传送出去。万一在广域网端口收到广播信息，也不会传进自己的局域网端口中。

既然网络都有发信端与收信端，用以标示信息发送者与信息接收者，除非对方使用一些特殊的封包封装方式或是使用防火墙对外连接，那么只要有人和我们的主机进行通信，就应该知道对方的地址，如果对方使用了防火墙，则至少也能够知道防火墙的地址。如果对方是通过一台 UNIX 主机和我们连接，则可以通过 ident 查到是谁连接的。

3.6.1　通信过程的记录设定

如果想要记录网络连接记录，可以使用 crontable：

```
netstat>>filename
```

UNIX 系统早已考虑到这一需求，因此在系统中有一个专门记录系统事件的 Daemon：syslogd，在 UNIX 系统的/var/adm 下面有两个系统记录文件 syslog 与 messages，其中，一个是一般系统的记录，一个是核心的记录。

系统的记录基本上都是由 syslogd（SystemKernelLogDaemon）来产生，而 syslogd 是由/etc/syslog.conf控制的。syslog.conf 以两个字段来决定要记录什么，以及记录到哪里。下面是一个 Linux 系统所附带的 syslog.conf 文件，这也是一个最标准的 syslog.conf 写法。

格式第一栏写“在什么情况下”以及“什么程度”。然后用 TAB 键转移到下一字段继续写“符合条件以后要做什么”。syslog.conf 文件只能用 TAB 来做各字段之间的分隔。

第一个字段包含了何种情况与程度，中间用小数点分隔。另外，星号代表了某一细项中的

所有选项。

对各种不同的情况,详细的设定方式以下面的字符串来决定。

- auth:关于系统安全与使用者认证方面。
- cron:关于系统自动排序执行(CronTable)方面。
- daemon:关于背景执行程序方面。
- kern:关于系统核心方面。
- lpr:关于打印机方面。
- mail:关于电子邮件方面。
- news:关于新闻讨论区方面。
- syslog:关于系统记录本身方面。
- user:关于使用者方面。
- uucp:关于 UNIX UUCP 方面。

上面是大部分的 UNIX 系统都会有的情况,而有些 UNIX 系统可能会再分出不同的项目。

并不是所有的信息都要记录,下面是各种不同的系统状况程度,依照轻重缓急排列。

- none:不要记录这一项。
- debug:程序或系统本身排错信息。
- info:一般性信息。
- notice:提醒注意性信息。
- err:发生错误。
- warning:警告性信息。
- crit:较严重的警告。
- alert:更严重的警告。
- emerg:非常严重的告警。

同样地,各种 UNIX 系统可能会有不同程度的表示方式。有些系统不另外区分 crit 与 alert 的差别,也有的系统会有更多种类的程度变化。在记录时,syslogd 会自动将所设定程度以及其上的程度都一并记录下来。

例如要系统去记录 info 等级的事件,则 notice、err.warning、crit、alert、emerg 等在 info 等级以上的也会一并被记录下来。把上面所列出的 1、2 项以小数点组合起来就是完整的“要记录哪些东西”的写法。例如 mail.info 表示关于电子邮件传送系统的一般性信息。auth.emerg 就是关于系统安全方面相当严重的信息。lpr.none 表示不要记录关于打印机的信息。另外有三种特殊符号可供应用:

- 惊叹号“!”:惊叹号表示不要记录目前这一等级以及其上的等级。
- 等号“=”:等号表示只记录目前这一等级,其上的等级不要记录。例如上面的例子,平常写下 info 等级时,也会把位于 info 等级上面的 notice、err.warning、crit、alert、emerg 等其他等级也记录下来。但若你写 =info 则就只记录 info 这一等级了。
- 星号“*”:星号代表某一细项中的所有项目。例如 mail.* 表示只要有关 mail 的,不管什么程度都要记录下来。而 *.info 会把所有程度为 info 的事件给记录下来。

3.6.2　查找记录

一般的 syslogd 都提供下列的 pipe 记录系统发生的事件：

- 一般文件。
- 指定的终端机或其他设备。
- 指定的使用者。
- 指定的远程主机。

1. 一般文件

可以指定好文件路径与名称，但是必须以目录符号“/”开始，系统才会知道这是一个文件。例如/var/adm/maillog 表示要记录到/var/adm 下面一个称为 maillog 的文件。如果之前没有这个文件，系统会自动产生一个。

2. 指定的终端机或其他设备

还可以将系统记录写到一个终端机或是设备上。若将系统记录写到终端机，则目前在使用该终端机的使用者就会直接在屏幕上看到系统信息(例如/dec/console 或是/dev/tty1. 可以拿一个屏幕来专门显示系统信息)。若将系统记录写到打印机，则会有一长条印满系统记录的纸(例如/dev/lp0)。

3. 指定的使用者

可以在这边列出一串使用者名称，则这些使用者如果正好上线的话，就会在他的终端机上看到系统信息。

4. 指定的远程主机

这种写法不将系统信息记录在连接本地的机器上，而记录在其他主机上。若是系统硬盘错误，或是硬盘摔坏了，系统记录就丢失了。因此，如果觉得某些情况下可能记录没办法存进硬盘里，可以把系统记录丢到其他的主机上。可以写下主机名称，然后在主机名称前面加上@符号(例如@ccunixl.variox.int，但被指定主机上必须要有 syslogd)。

在以上各种记录方式中，都没有电子邮件这项。因为电子信件要等收件者去收信才看得到，有些情况可能是很紧急的，没办法等待。

以上就是 syslog 各项记录程度以及记录方式的写法，可以依照自己的需求记录下自己所需要的内容。但是这些记录都是一直堆上去的，除非将文件自行删除掉，否则这些文件就会越来越大。有的人可能会在 syslog.conf 里面写下如下代码：

```
*.* /var/log/everything
```

此时所有的情况都被记录下来了。如果系统出问题了，可能要从好几十兆字节甚至几百兆字节的文字中找出问题所在，这样可能对我们一点帮助都没有。因此，以下两点可以帮助我们快速找到重要的记录内容：

- 定期检查记录

每周或是更短的时间看一次记录文件。如果需要，可将旧的记录文件备份，可以写入 cploglog.1，cploglog.2... 或是 cploglog.971013，cploglog.980101... 等，将过期的记录文件依照流水号或是日期保存起来，未来考察时也比较容易。

- 只记录有用的东西

不能记录下 * . * 然后放在一个文件中,这样会导致文件太大,要找信息时根本无法马上找出来。有人在记录网络通信时,连谁去 ping 他的主机都记录。除非是系统已经遭到很大的威胁,否则这样的小事可以不用记录。这样做可以提升系统效率以及降低磁盘用量,当然也节省了时间。

3.6.3 地理位置的追踪

如何查出入侵者的地理位置呢? 在专线接入网络环境中,入侵者一定和网络提供单位有着密切的关系。因为对于区域网络,距离绝对不出几公里。即使是拔号,也很少有人会花大笔钱去拨外地甚至国外的拔号服务器。因此,只要查出连接的单位,入侵者必然离连接单位不远。

但拔号式的网络就比较难于追踪了。现在有很多的小时卡、记点卡等,不需申请,账号密码就直接附在卡片上面。用户只要买了固定的小时数,不需向 ISP 提出申请,就可以按照卡片上的说明自行拔号上网。这样 ISP 就根本无从得知是谁在用他们的网络。也就是说,以小时卡提供拔号服务给拔号使用者带来相当大的便利,但却是系统安全的大敌,网络黑客的捷径。

3.6.4 来电显示

ISDN 的 CallerID 可以显示对方的电话号码,但是 CallerID 依然有失效的时候。那么,CallerID 可以显示出哪些号码呢? 要显示来电话方号码的前提是,对方必须是通过数字交换机打过来,在某些地区仍然使用机械式交换机,或者使用集团程控的地区,就无法显示出来电号码了。

3.6.5 使用 IP 地址和域名

(1) 使用 IP 地址和域名查找入侵者位置

如果电话查不出来,可以查 IP 地址。IP 地址的使用必须向 InterNIC 登记,而域名要向当地直属的网络管理中心登记。在 Internet 上的网络管理中心共有三层(单位性质一定为 NET):国际等级、洲际等级、国家等级。

1) 国际等级。国际等级只有 InterNIC 一个,全球各国的 NIC 以及洲际 NIC 均由其管理。其网址是:http://www.internic%20.net/。

2) 洲际等级。InterNIC 并不直接管理整个 Internet,其下的网络资源会再做分区。例如亚太洲际网络管理中心(Asian-PacificNIC, APNIC,位于日本)管理亚太地区的 IP 地址,其网址是:http://www.apnic.net/。

3) 国家等级。域名后面的由当地国家的 NIC 管理,惯例是两位国码加上 NIC 就是该国 NIC 的名称。

但是由于 InterNIC 位于美国,因此美国的域名由 InterNIC 直辖。有一个特别的例外是挂 .mil 的美国军方网络的信息是由 ddn.mil(美国军事防卫网络)来管理,不由 InterNIC 管理,当获得某个域名或是 IP 地址后,可以使用 whois 来查出信息,语法如下:

Whois-h<whois 服务器><查询对象>

whois 也可以使用下列语法:

whois<查询对象>@<whois 服务器>

(2) 域名命名的三种情况

虽然同样是域名,可能会遇到三种不同的命名情况。在许多国家 * .edu. * 是由 NIC 以外的单位管理(如教育部),而属性也不一定是三个字母,甚至没有属性。在判断单位性质时需要多加注意,以免找不到信息。

- 标准国码 + 三码属性码:普遍使用于欧美国家以及部分东南亚国家,如 * .com.cn、* .edu.cn等。
- 标准国码 + 二码属性码:典型的使用国家为日本,例如在日本公司属性为 co,社团属性为 or。
- 仅有标准国码,未有任何属性码:澳洲的主机均为仅有 * .au 的主机名称,而没有任何其他的 com 或 co。

(3) 通过域名查找连接单位信息

按照 Internet 上的惯例,由 whois 服务来查询连接单位的登记信息,whois 本来应该是用来查某人的电话或是其他信息的,但是在 NIC 方面是用来查出连接单位的电话以及住址、技术联络人等。符合该 NIC 管理权限的单位信息会存放于该单位的 whois 主机中,惯例是 whois + NIC 名称 + net。例如亚太地区网络管理中心 whoisserver 为 whois.apnic.net。

知道某台主机的域名以后,可以依照下面顺序查出连接单位的电话住址等信息。先看有没有国码。没有国码的,向 whois.internic.net 询问;有国码的,向 whois. 国码 nic.net 询问。

(4) 由域名查出信息

如果能从 nslookup 查出某一 IP 地址的地址,则可以直接向当地 NIC 查出入侵者网络的信息。

(5) 只有 IP 地址的查法

若发现 168.95.109.222 有人入侵,假设不知道这是 HiNet 的网络,而这个 IP 地址也没有域名的话,则须先将 IP 地址分等级,再向 InterNIC 查询 。

3.6.6　Web 欺骗的攻击和策略

Web 欺骗技术是一种在 Internet 上使用的针对 WWW 的攻击技术,这种攻击方法会泄露某人的隐私或破坏数据的完整性,危及到使用 Web 浏览器的用户,包括使用 Netscape Navigator 和 Internet Explorer 的用户。

1. Web 面临的安全威胁

越来越多的用户通过使用 WWW 获取信息,进行各种交流和交易,然而很少有人意识到当他们使用 WWW 的时候,却有少数攻击者正在窥视着他们的一举一动,利用这一领域人们知识的缺乏和疏忽进行一些捣乱和破坏。主要表现在下面三个方面:

- Web 页面的欺诈。
- CGI 欺骗。
- 错误和疏漏。

2. Web 攻击的行为和特点

Web 欺骗是指攻击者建立一个使人相信的 Web 页站点的拷贝,这个 Web 站点拷贝就像真的一样,具有所有的页面和连接。然而攻击者控制了这个 Web 站点的拷贝,被攻击对象和

真实的 Web 站点之间的所有信息流动都被攻击者控制。

Web 攻击技术使得攻击者可以创建整个 Web 站点的“影子拷贝”。用户访问影子 Web 会经过攻击者的机器，这样攻击者就可以监视被攻击者对象的所有信息，包括账号和口令以及其他的一些信息。攻击者既可以假冒成用户给服务器发送数据，也可以假冒服务器给用户发送假冒的消息。总之，攻击者可以监视和控制整个过程。

3.7 练习题

一、选择题

1. 入侵检测的过程不包括下列哪个阶段(　　)。

A. 信息收集　　B. 信息分析

C. 信息融合　　D. 告警与响应

2. 在入侵分析模型中，第一阶段的任务是(　　)。

A. 构造分析引擎　　B. 进行数据分析

C. 反馈　　D. 提炼

3. 异常检测依靠的一个假定是(　　)。

A. 一切网络入侵行为都是异常的。

B. 用户表现为可预测的、一致的系统使用模式。

C. 只有异常行为才有可能是入侵攻击行为。

D. 正常行为和异常行为可以根据一定的阈值来加以区分。

二、简答题

1. 简述入侵分析的目的。
2. 简述入侵分析需要考虑的因素。
3. 简述误用入侵检测方法的优点。
4. 简述异常入侵检测不同于误用入侵检测的特点。
5. 简述告警与响应的作用。
6. 简述如何实现简单的入侵追踪。

第 4 章　入侵检测系统的性能指标和评估标准

本章导读：

本章介绍入侵检测系统的性能指标和评估标准。首先介绍影响入侵检测性能的参数；接着介绍评价检测算法性能的测度和评价入侵检测系统性能的标准；接下来简要介绍关于网络入侵检测系统的测试评估；然后介绍测试评估的内容；接着对测试环境和测试软件进行了介绍；然后简要介绍了用户评估标准；最后对入侵检测评估现状进行了分析。

当一个网络入侵检测系统的设计和实现完成后，我们关心的问题就是 IDS 是否达到了设计目标。那么如何去测试评估 IDS，评估 IDS 需要考虑哪些性能指标呢？本章主要介绍网络入侵检测系统的性能指标和测试评估，包括评估标准、评测方法、相应的测试工具和 IDS 的评估现状。

4.1　影响入侵检测系统性能的参数

分析入侵检测系统的性能时，应重点考虑检测的有效性和效率。有效性是指研究检测机制的检测精确度和系统报警的可信度。效率则是从检测机制处理数据的速度以及经济角度来考虑，侧重检测机制性能价格比的改进。本节只从检测有效性的角度，对检测系统的检测性能及影响性能的参数进行分析。我们期望检测系统能够最大限度地把系统中的入侵行为与正常行为区分开来。这就涉及到入侵检测系统对系统正常行为（或入侵行为）的描述方式、检测模型与检测算法的选择。如果检测系统不能够精确地描述系统的正常行为（或入侵行为），那么系统必然会出现各种误报。如果检测系统把系统的“正常行为”作为“异常行为”进行报警，这种情况就是虚警（false positive）。如果检测系统对部分针对系统的入侵活动不能识别、报警，这种情况被称作系统的漏警（false negative）现象。显然，过多的虚警必然会降低检测系统报警信息的可信度，甚至使得检测系统根本不实用。而漏警的危害性更大。基于异常检测的入侵检测系统是根据系统正常行为的特征轮廓所进行的排他性检测，虚警现象是影响这类系统实用化的重要障碍。这里，我们通过贝叶斯理论来分析基于异常性检测的入侵检测系统的检测率、虚警率与报警可信度之间的关系，并在此基础上分析它们对异常性检测算法性能的影响。

事实上，入侵检测问题可看作是一个简单的二值假设检验问题。为便于分析，我们首先给出一系列相关的定义和符号：

假设 I 与 $\neg I$ 分别表示入侵行为和目标系统的正常行为，A 代表检测系统发出了入侵报警，$\neg A$ 表示检测系统没有报警。

检测率：指被监控系统受到入侵攻击时，检测系统能够正确报警的概率，可表示为 $P(A|I)$。通常利用已知入侵攻击的实验数据集合来测试入侵检测系统的检测率。

虚警率：指检测系统在检测时出现虚警的概率，可表示为 $P(A|\neg I)$。可利用已知的系统

正常行为实验数据集，通过系统仿真获得检测系统的近似虚警率。

另外，概率 $P(\neg A|I)$代表检测系统的漏警率，$P(\neg A|\neg I)$则指目标系统正常(没有入侵攻击)的情况下，检测系统不报警的概率。显然有：

$$P(\neg A|I)=1-P(A|I)\quad P(\neg A|\neg I)=1-P(A|\neg I)$$

而实际应用中，我们则主要关注一个入侵检测系统的报警结果是否能够正确地反映目标系统的安全状态。下面的两个参数则从报警信息的可信度方面考虑检测系统的性能：

$P(I|A)$给出了检测系统报警信息的可信度，即检测系统报警时，目标系统正受到入侵攻击的概率。该参数小于 1 时，检测系统存在虚警现象。

$P(\neg I|\neg A)$则给出了检测系统没有报警时，目标系统处于安全状态(没有受到入侵攻击)的可信度。该参数小于 1 时，检测系统存在漏警现象。

显然，为使入侵检测系统更有效，我们期望系统的这两个参数的值越大越好。下面，根据贝叶斯定理给出这两个参数的计算公式：

$$P(I|A)=\frac{P(I)P(A|I)}{P(I)P(A|I)+P(\neg I)P(A|\neg I)} \tag{4-1}$$

同理：

$$\begin{aligned}P(\neg I|\neg A)&=\frac{P(\neg I)P(\neg A|\neg I)}{P(\neg I)P(\neg A|\neg I)+P(I)P(\neg A|I)}\\&=\frac{P(\neg I)(1-P(A|\neg I))}{P(\neg I)(1-P(A|\neg I))+P(I)(1-P(A|I))}\end{aligned}$$

根据上面的分析，针对给定的实验数据或具体环境，可以通过统计以及系统仿真获得 $P(I)$、$P(\neg I)$、$P(A|I)$以及 $P(A|\neg I)$的先验概率，继而计算出这两个检测系统可用性参数的评估结果。由于入侵检测系统是根据其检测结果，以对入侵行为进行报警的形式通知被监控系统，所以在下面的分析中，主要关注报警信息的可信度 $P(I|A)$与检测系统性能的关系。

由于对目标系统的入侵攻击事件是一个随机事件，一般与检测系统无关，因而在分析检测系统的性能时，采用同样的实验数据集和实验环境，这样就可以通过概率获得关于入侵事件的先验概率 $P(I)$。又因为在所有被监控系统的行为中，入侵行为出现的概率一般很小，即：

$$P(I)\ll P(\neg I)$$
$$P(I)+P(\neg I)=1$$

所以在式(4-1)中，分母将主要受虚警率的影响。为了便于分析，假定目标系统在某一时间段内遭受入侵攻击的概率为 $P(I)=0.00001$，并在图 4-1 中给出了检测系统的报警信息可信度与系统检测虚警率、检测率之间的关系。由图 4-1 中曲线可知：在给定检测率的条件下，报警信息的可信度将随着检测系统虚警率的增大而减小。而在给定虚警率的条件下，报警信息的可信度将随着检测率的增大而增大。虽然当 $P(A|I)=1$ 且 $P(A|\neg I)=0$ 时，检测的结果最可信(每次报警都预示着系统受到了入侵，且每次入侵都能够被检测出来)。但这只是最理想的情况，在实际的系统中很难达到。

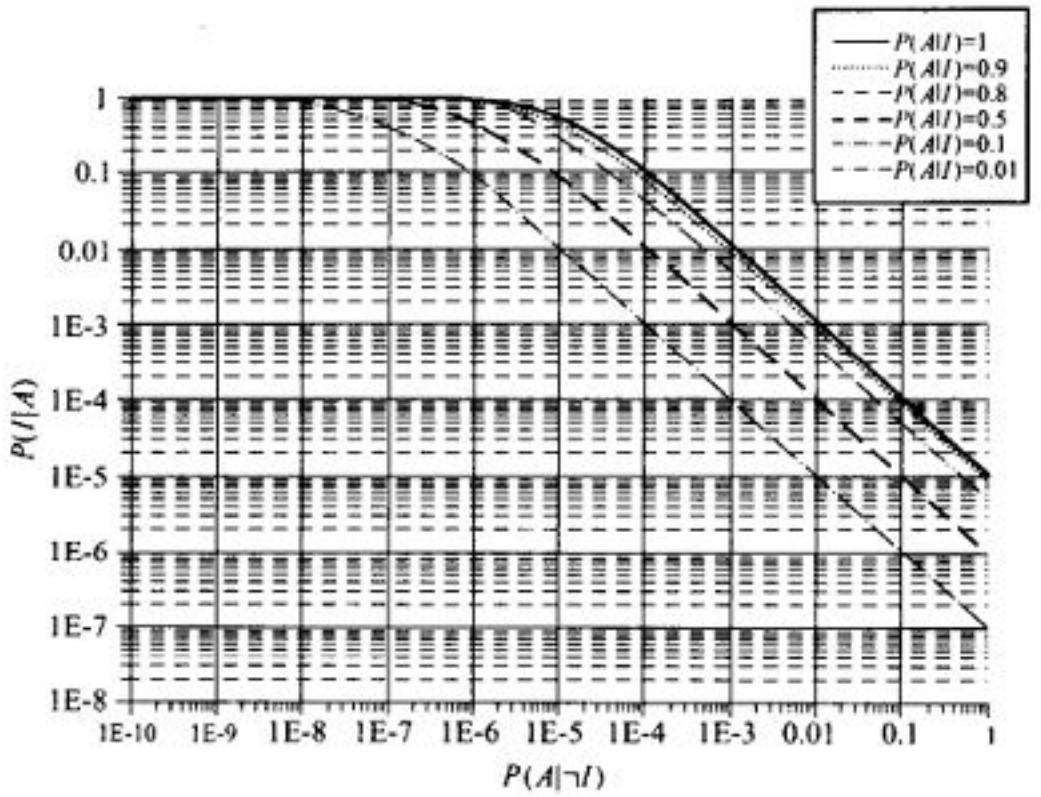

图 4-1　检测系统的报警信息可信度与虚警率、检测率之间的关系

4.2　评价检测算法性能的测度

通过对检测系统报警可信度的分析可知，设计一个好的入侵检测系统，不仅要尽力提高系统的检测率，而且要尽可能地降低系统的虚警率，提高系统报警的可信度。在一个具体的异常性检测系统中，由于描述系统行为的特征轮廓的不完善，较高的检测率可能也同时预示着较高的虚警率。可利用检测率随虚警率的变化曲线来评价检测系统的性能，而这条曲线称为接收器特性(Receiver Operating Characteristic，ROC)曲线。根据一个检测系统在不同条件(在允许范围内变化的阈值，例如异常检测系统的报警门限等参数)下的虚警率和检测率，分别把虚警率和检测率作为横坐标和纵坐标，就可做出对应于该检测系统的 ROC 曲线。显然，ROC 曲线与检测系统的检测门限具有对应的关系。当检测系统没有影响系统性能的可调参数时，那么该检测系统的 ROC 曲线就退化成了一个点。

假设有三种采用不同检测算法(或检测机制)的入侵检测系统 A、B、C，分别做出它们的 ROC 曲线，就可以得到一组 ROC 曲线簇，如图 4-2所示。可以很容易地看出，系统 C 始终优于系统 B，系统 A 则代表最坏的情况，相当于对入侵行为没有识别能力。如果一个检测系统的检测性能比系统 A 还差的话(曲线在 A 曲线的下方)，只需把检测系统的判断结果取反，就可以使新的检测系统的性能高于系统 A，达到改善系统检测性能的目的。从这个角度来看，检测系统的性能图中，$y = x$ 可以作为系统的性能基

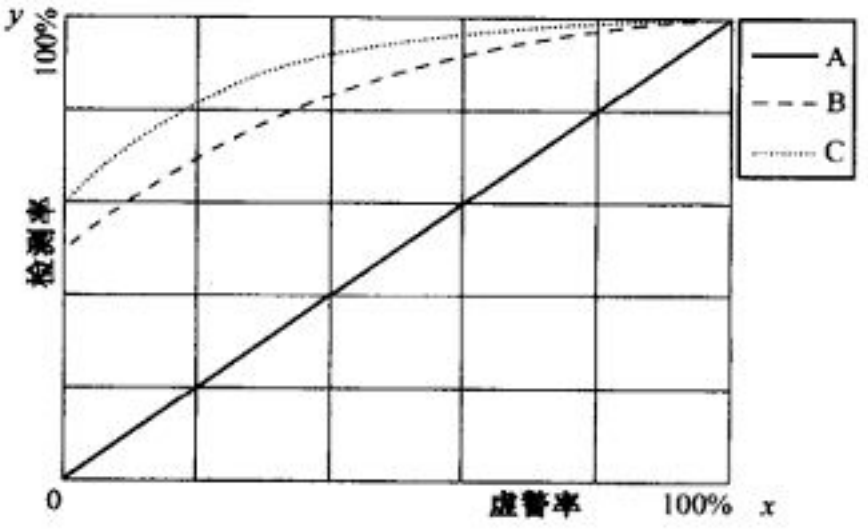

图 4-2　表示不同检测系统性能的 ROC 曲线簇

准,所有入侵检测系统的性能都应比它所代表的系统好。

图中有三个需要关注的坐标点:(0,0)、(1,1)以及(0,1)。(0,0)坐标点表示一个检测系统的报警门限过高,从而根本就检测不出入侵活动的情况。(1,1)坐标点则表示检测系统的报警门限为0时,检测系统把被监控系统的所有行为都视为入侵活动的情况。至于坐标点(0,1)则代表了一个完美的检测系统,能够在没有虚警的条件下,检测出所有的入侵活动,显然这是一个理想的情况。

根据检测率与虚警率的ROC曲线,不仅可以确定检测系统在给定虚警率门限时的检测性能,还可以通过调整检测系统的检测门限参数对检测率和虚警率进行均衡,使它们达到我们可接受的程度。调整过程中,若我们关注的重点在于降低虚警率,可以通过对虚警率坐标轴的局部细化,重点考虑虚警率对检测性能的影响。若我们更关注提高检测率,希望能检测出所有的入侵活动,那么在调整性能参数时,就要首先对代表系统检测率的坐标轴进行细化。

我们知道检测算法是检测系统的核心,利用同一检测系统模型,通过采用不同的检测算法,可以得到不同的入侵检测系统。因而,入侵检测机制与检测算法的性能可通过ROC曲线进行评价。

4.3 评价入侵检测系统性能的标准

虽然入侵检测及其相关技术已有很大的进展,但关于入侵检测系统的性能检测及其相关评测工具、标准以及测试环境等方面的研究工作还不够。根据Porras等的研究,下面给出评价入侵检测系统性能的三个因素:

- 准确性(Accuracy):指入侵检测系统能正确地检测出系统入侵活动。当一个入侵检测系统的检测不准确时,它就可能把系统中的合法活动当作入侵行为并标识为异常(虚警现象)。
- 处理性能(Performance):指一个入侵检测系统处理系统审计数据的速度。显然,当入侵检测系统的处理性能较差时,它就不可能实现实时的入侵检测。
- 完备性(Completeness):指入侵检测系统能够检测出所有攻击行为的能力。如果存在一个攻击行为,无法被入侵检测系统检测出来,那么该入侵检测系统就不具有检测完备性。由于在一般情况下,我们很难得到关于攻击行为以及对系统特权滥用行为的所有知识,所以关于入侵检测系统的检测完备性的评估要相对困难得多。

在此基础上,Debar等又增加了两个性能评价测度:

- 容错性(Fault Tolerance):入侵检测系统自身必须能够抵御对它自身的攻击,特别是拒绝服务攻击(Denial-Of-Service)。由于大多数入侵检测系统是运行在极易遭受攻击的操作系统和硬件平台上,这就使得系统的容错性变得特别重要,在设计入侵检测系统时必须考虑。
- 及时性(Timeliness):及时性要求入侵检测系统必须尽快地分析数据并把分析结果传播出去,以使系统安全管理者能够在入侵攻击尚未造成更大危害以前做出反应,阻止攻击者颠覆审计系统甚至入侵检测系统的企图。和上面的处理性能因素相比,及时性要求更高。它不仅要求入侵检测系统的处理速度要尽可能地快,而且要求传播、反应检测结

果信息的时间尽可能少。

4.4　网络入侵检测系统测试评估

测试评估入侵检测系统非常困难，涉及到操作系统、网络环境、工具、软件、硬件和数据库等技术方面的问题。由于入侵检测技术是新技术，商业的 IDS 新产品周期更新非常快，IDS 目前没有工业标准可参考来评测。市场化的 IDS 产品很少去说明如何发现入侵者和日常运行所需要的工作及维护量。IDS 厂商考虑到商业利益不会公布检测算法，竞争对手之间互相隐藏攻击签名，当然这也为了防止攻击者确切得知签名的工作机制。判断 IDS 检测的准确性只有依靠黑箱测试。另外，测试 IDS 需要构建网络和操作系统环境以及网络通信流量样本数据，系统、进程、文件使用和用户行为的轮廓也需要测试数据。由于不断变化的入侵攻击的情况，用户和厂商需要维护多种不同类型的信息才能保证 IDS 能够检测到可疑事件。这些信息包括：

- 正常和异常情况下用户、系统和进程行为的轮廓。
- 可疑通信量模式字符串，包括已知的入侵攻击签名。
- 对各种异常和攻击进行响应所需要的信息。

IDS 厂商来维护这些信息，但并不是全部。IDS 的管理员根据安全手册要经常更新这些信息。现在依靠单独的技术不足以维护网络安全，一个组织或部门需要合格的技术人员来评估、选择、安装、操作和维护 IDS。目前，缺乏计算机安全的入侵分析员和网络系统员，一般用户都希望买到具有智能分析能力的入侵检测产品。对于那些缺少技术特别是有关安全方面的用户来说，更是希望如此。但是，每天所发生的许多攻击和扫描，只有训练有素的分析员，而且还要借助专家级的工具才能够检测到。安全专家常以手工方式才能抓住入侵者。由此看来，要实现完全自动化进行入侵检测处理尚需努力。面对这样的现实，用户要根据自己需求，担当一定的风险来选择 IDS 产品。

IDS 的测试评估者主要来自开发者、第三方、入侵者和最终用户。开发者和第三方的测试活动在 IDS 产品早期进行，测试的环境也有限制，而入侵和最终用户的测试评估是在实际应用环境下进行的。因此，最终能否经得起考验，关键还要看入侵者和最终用户的测试评估效果。

IDS 只能测试已知攻击。在许多环境中，已知攻击的发生率要比新攻击高，因为已知的攻击方法已广泛公布，而大部分的人了解新的攻击细节就可能少得多。一般来说，新的攻击常类似于已知的攻击，若 IDS 能够检测到大部分已知的攻击方法，那么就有可能检测到新的攻击。模拟入侵者来实施攻击测试面临的困难是只能掌握已公布的攻击，而无法得知新的攻击方法。不过，可以通过分类选取测试例子尽量覆盖不同种类的攻击，同时不断收集新的入侵攻击信息，更新网络仿真测试的数据以适应新的情况。然而即使测试没有发现 IDS 的所有潜在弱点，也不能说明 IDS 是一个完备系统。若运行 IDS 和监视管理 IDS 软件的计算机系统被入侵者控制，则入侵者就能够操作 IDS 程序本身。假如入侵者又知道 IDS 数据库中的入侵签名知识，从而针对入侵数据库中没有出现入侵签名的攻击进行尝试，IDS 就不会发现。因此，应当保障入侵数据库的保密性。一般情况下，IDS 的测试环境由测试平台控制器、正常合法使用用

户、攻击者、服务器和 IDS 系统组成。如图 4-3 所示。服务器用来运行各种网络服务的计算机，为正常客户提供服务访问，还可当作攻击者的目标。服务类型表示服务器提供的网络服务。若 IDS 是基于主机型，则 IDS 运行在服务器上，或者服务器将其产生的审计信息发送到其他机器上处理。IDS 不必在一台服务器上运行，可以在一台专用的计算机上运行，这要根据 IDS 的类型来安排。测试环境另外的组成部分是模拟用户和攻击者，通过软件方式产生正常用户网络流信息和攻击者所生成的信息流。

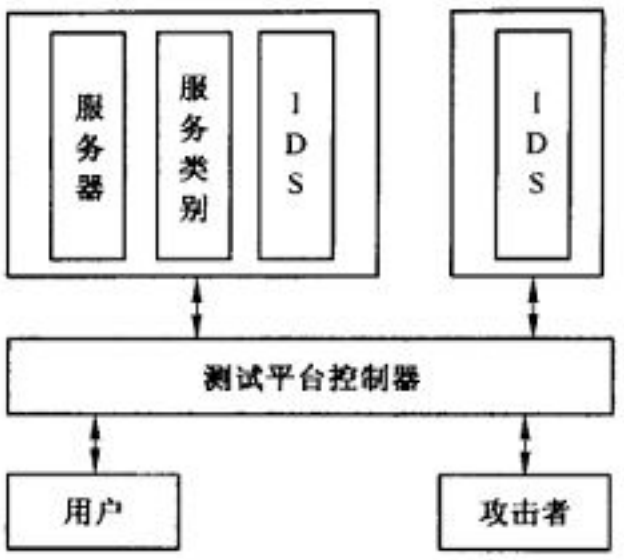

图 4-3 IDS 测试环境组成示意图

4.5 测试评估内容

IDS 通用的检测方法是误用检测和异常检测。大多数 IDS 都包含这两种检测模块。误用检测依据两个关键部分：入侵签名数据库和模式匹配算法。误用检测失效的原因有：

- 系统活动记录未能为 IDS 提供足够的信息用来检测入侵。
- 入侵签名数据库中没有某种入侵攻击签名。
- 模式匹配算法不能从系统活动记录中识别出入侵签名。

异常检测以入侵者的行为不同于典型用户的行为为基础，通过构建轮廓模板来描述用户的行为特征，如登录时间、CPU 的使用、磁盘使用、访问敏感文件等，形成一种可数量化的指标，入侵检测系统持续地根据系统或用户行为维护这个指标，当某个这种指标突然越出一个界限，就认为异常行为发生了，并由此检测到入侵行为。异常检测局限性取决于用户一致性。在某些环境下，合法的用户可能会频繁改变自己的行为。在这种情况下，就难以建立起这些用户的轮廓模板。异常检测模块失效的原因通常有：

- 异常阈值定义不合适。
- 用户轮廓模板不足以描述用户的行为。
- 异常检测算法设计错误。

IDS 的评估涉及到入侵识别能力、资源使用状况、强力测试反应等几个主要问题。入侵识别能力是指 IDS 区分入侵行为和正常行为的能力。资源使用状况是指 IDS 消耗多少计算机系统资源，以便将这些测试的结果作为 IDS 运行所需的环境条件。强力测试反应是指测试 IDS 在特定的条件下对所受影响的反应，如负载加重情形下 IDS 的运行行为。下面就 IDS 的功能、性能以及产品可用性三个方面作一些具体讨论。

4.5.1 功能性测试

功能测试出来的数据能够反映 IDS 的攻击检测、报告、审计、报警等能力。

1. **攻击识别**

以 TCP/IP 协议攻击识别为例，攻击识别可以分成以下几种。

- 协议包头攻击分析的能力：IDS 系统能够识别与 IP 包头相关的攻击能力，如 LAND 攻击。其攻击方式是通过构造源地址、目地地址、源端口、目的端口都相同的 IP 包发送，这样导致 IP 协议栈产生 progresive loop 而崩溃。
- 重装攻击分析的能力：IDS 能够重装多个 IP 包的分段，并从中发现攻击。常见的重装攻击是 Teardrop 和 Ping of Death。Teardrop 通过发多个分段的 IP 包而使得当重装包时，包的数据部分越界，进而引起协议和系统不可用。Ping of Death 是将 ICMP 以多个分段包(碎片)发送，而当重装时，数据部分大于 65535B(字节)数，从而超出 TCP/IP 协议所规定的范围，引起 TCP/IP 协议栈崩溃。
- 数据驱动攻击分析能力：IDS 能够分析 IP 包数据的具体内容。例如 HTTP 的 phf 攻击。phf 是一个 CGI 程序，允许在 WEB 服务器上运行。由于 phf 处理复杂服务请求程序的漏洞，使得攻击者可以执行特定的命令，从而可以获取敏感的信息或者危及到 Web 服务器的使用。

2. 抗攻击性

可以抵御拒绝服务攻击，对于某一时间内的重复攻击，IDS 报警能够识别并能抑制不必要的报警。

3. 过滤

IDS 中的过滤器可方便地设置规则，以根据需要过滤原始的数据信息，例如网络上的数据包和审计文件记录。一般要求 IDS 过滤器具有下面的能力：

- 可以修改或调整。
- 创建简单的字符规则。
- 使用脚本工具创建复杂的规则。

4. 报警

报警机制是 IDS 必要的功能，例如发送入侵警报信号和应急处理机制。

5. 日志

- 保存日志的数据能力。
- 按特定的需求说明，日志内容可以选取。

6. 报告

- 产生入侵行为报告。
- 提供查询报告。
- 创建和保存报告。

4.5.2　性能测试

性能测试在各种不同的环境下，检验 IDS 的承受强度，主要的指标有下面几点：

- IDS 引擎器的吞吐量：IDS 在预先不加载攻击标签的情况下，IDS 处理原始检测数据的能力。
- 包的重装：测试的目的就是评估 IDS 的包的重装能力。例如，为了测试这个指标，可通过 Ping of Death 攻击，IDS 的入侵标签库只有单一的 Ping of Death 标签，这时来测试 IDS 的响应情况。

- 过滤的效率:测试的目标就是评估 IDS 在攻击的情况下过滤器的接收、处理和报警的效率。这种测试可以用 LAND 攻击的基本包头为引导,这种包的特征是源地址等于目标地址。

4.5.3 产品可用性测试

评估系统的用户界面的可用性、完整性和扩充性。支持多个平台操作系统,容易使用且稳定。

4.6 测试环境和测试软件

4.6.1 测试环境

美国的 Lincoln 实验室和 Rome 实验室做了 IDS 的测试评估工作。Lincoln 实验室设计了一个离线的网络 IDS 测试环境,如图 4-4 所示。其方法是通过使用大量的含有各种各样攻击的网络流量样本和审计数据样本作为 IDS 系统的输入,以此验证 IDS 的能力,并计算出虚警率和检测率。

Rome 实验室则设计一个实时的网络环境以用作评测 IDS,如图 4-5 所示。其方法是将 IDS 放在实际的网络环境中,以验证 IDS 在实时情况下的响应状况和适应性。

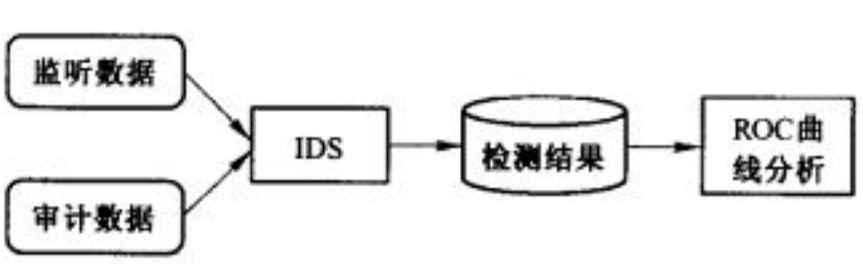

图 4-4 Lincoln 实验室的 IDS 测试环境

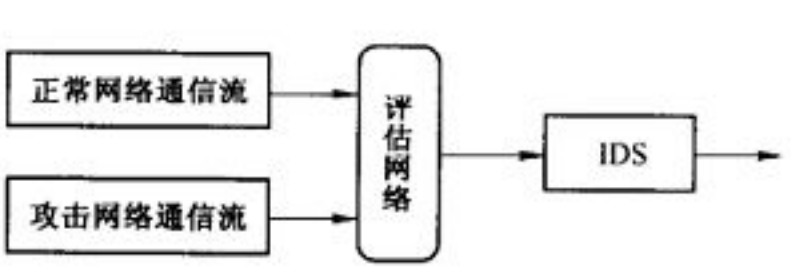

图 4-5 Rome 实验室的 IDS 测试环境

为了较好地测试 IDS,针对不同的测试目标,一般构建专用的网络环境,图 4-6 是测试 IDS 功能的网络配置图。网络负载模拟器用来产生网络通信流,攻击目标则专为攻击测试设定,IDS 探测器和 IDS 管理中心构成入侵检测系统用于监测攻击测试的整个过程。攻击者负责产生各种各样的攻击网络通信流以检验 IDS 的检测功能。攻击者可预先构建攻击脚本,然后自动去执行。例如,通过 TCPDUMP 工具将含有攻击活动的网络上流动的信息"复制"一份,然后用其他软件重放。

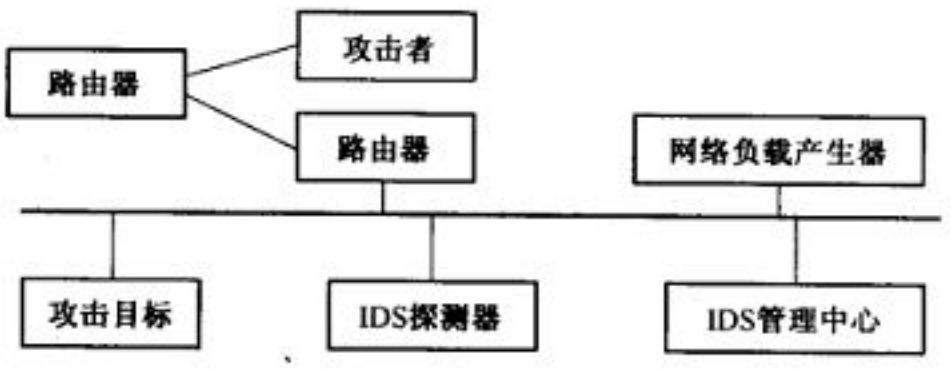

图 4-6 IDS 功能测试配置图

性能测试与功能测试有所不同，因而测试的网络环境以局域网为主，这样可以尽最大可能的产生大的网络流量，避免路由器等设备对网络流量的约束。

4.6.2　测试软件

IDS 测试软件协助测试员模拟实际环境，自动输入测试数据以及训练检测模型，减少手工劳动量，提高测试效率。其软件一般功能如下：

- 仿真用户操作。生成各种各样的用户正常使用模式，这些模式对于基于异常检测模型和误用检测模型的入侵检测系统来说都是重要的。
- 模拟入侵。入侵有串行或并行两种形式。串行入侵方式是入侵者从一台终端或计算机上单独发布一系列攻击命令。并行方式是一个或多个入侵者从若干台终端或计算机上同时发布一系列攻击命令操作。

目前，仿真网络用户的行为有通用会话生产工具(Generic session generation tool)、测试软件包(Using test suites)和录制实际数据重放(mcording live data)三种方法。通用会话生产工具方法是基于有限自动机来实现一个用户所有可能的操作。可以用随机 Petri 网络建立用户模型。

操作系统开发商自带测试软件包是比较简单的模拟方法，通常用于测试评估操作系统的服务性能和应用服务软件是否按设计说明来实现。不足的是这些测试不能给出用户常常进行什么样的操作，但是可以让我们知道系统对正常请求的响应行为。录制实际数据重放方法是通过记录各种各样的用户正常活动的数据，然后在测试平台上重放用户的活动过程。这种方法要求有足够多的用户活动记录，以便模拟服务器超载时的情形。同时，应区分训练和测试阶段使用的用户活动记录。训练阶段只用到用户活动记录的一部分来建立异常模型，而测试阶段使用其他部分来验证 IDS 所创建的用户使用模型。

模拟攻击是测试软件的一个必不可少的功能，通过运行攻击来验证 IDS 是否能够检测到这些攻击。可收集网上已公布的弱点开发相应的攻击脚本，然后直接运行测试。一些黑客杂志也常讨论攻击方法，并有源程序。但是，某些攻击方法与环境相关，例如缓冲区溢出。此时，我们将受环境约束的攻击以重放方式实现。事先收集这些攻击的网络信息流，然后通过重放软件将其输送到测试网络上获得新的入侵签名和一些攻击特征信息流。

1. nidsbench 测试软件包

nidsbench 是由 Anzen 公司开发的一套 IDS 的测试软件包，包括 tcpreplay 和 fraqrouter 两个部分，tcpreplay 的功能是将 tcpdump“复制”的网络数据包重放，以还原当时网络的实际运行状态。fraqrouter 的功能是通过检测一系列躲避 IDS 检测的攻击以测试 IDS 的正确性和安全性。

2. California 大学的 IDS 测试平台

California 大学开发了一个入侵检测系统软件测试平台，通过它可以实现模拟入侵，以测试 IDS 的有效性。该软件平台是使用 Tcl-DP 工具开发实现的。共包含四组命令：

- 基本的会话命令集：一般是常用的几个命令。如 telnet、login、ftp 这些命令用来仿真入侵者的基本操作。
- 同步命令集：提供按指定的要求产生某个事件。如 sys _ connect _ server 命令被用作让某个客户与指定服务端口建立连接。

- 通信命令集:该命令集实现并发的进程彼此通信,模拟一组用户一起工作。并发模拟脚本中的发送和接收通信命令用于进程之间的数据交换。
- 记录重放命令集:该命令集的功能是记录用户会话期间的操作命令序列,然后再重放这些记录。

除此之外,麻省理工学院 Lincoln 实验室开发了非实时的 IDS 性能评估工具,该工具能够动态地加速重放大量的数据,迅速产生所需测试数据。IBM 公司 Zurich 研究实验室也开发了一套 IDS 测评工具。此外,还有一些黑客工具软件也可用作 IDS 测试。就 IDS 测试软件总体状况来看,目前正处于发展期间。

4.7 用户评估标准

用户评估 IDS 涉及多种因素,本节从用户的用度,来检验 IDS 是否满足用户的需求。简单地罗列了相关的问题要点。这也是 IDS 走向市场化产品所要考验的问题。

1. 产品标识

- 生产厂商或公司。
- 产品版本号。
- IDS 的类型(基于主机或是基于网络)。
- IDS 监测日志或是网络包,或者两者都有之。
- IDS 的运行方式:独立还是 Client/Server 结构。
- IDS 产品形式是硬件、软件还是软硬结合。

2. IDS 文档和技术支持

- 全面、清晰、准确、组织良好的产品文档。
- 用户手册。
- 电子文档。
- 产品培训。
- 技术支持,涉及到服务效率、响应速度、价格等。

3. IDS 的功能

- 产品与现有系统的集成。
- 即插即用,还是要求改变配置或调整现有的系统。
- 所兼容和支持的软件平台,例如操作系统(UNIX,NT,OS/2)。
- 与其他的 IDS、防火墙和支持工具是否容易集成。
- IDS 运行需要的网络拓扑结构。
- 支持管理的方式,http,Telnet,SNMP,SNMP3,DECnet,远程串行终端。
- 管理协议(SNMP,SNMPv3)。
- IDS 企业级管理平台,与现有网络管理平台(HP OpenView,Solstice SunNet Manager,Tivoli Netview)的结合。
- 支持物理网络拓扑结构,Ethernet,Fast Ethernet,token ring,asynchronous transfer mode,FDDI)。

- 产品是否开放源代码。
- 应用程序编程 API 形式，怎样扩展。
- 和弱点扫描器工具集成。
- 用户和系统透明度。
- 支持的网络协议(IP，IPX，Appletalk，XNS，SNA，X25)。
- 安装和运行 IDS 时是否需要超级用户身份或修改内核。
- 监测的应用服务列表。
- 在管理控制平台断开、去掉、做拒绝服务测试时，检测能否持续。

4. **IDS 报告和审计**

- 灵活、扩展、配置报告方法。
- 每个用户、主机、站点、服务格式。
- 检测数据是否输入到外部数据系统。
- 尽可能实时通知(E-mail，SNMP，traps，page)。
- 所支持的审计方式(硬拷贝，远程审计)。
- 是否提供审计分析工具还是从第三方得到。
- 是否提供软件自动生成报告。
- 误报警和漏检率。

5. **IDS 检测与响应**

- 基于网络攻击脚本保护(地址欺骗，序列号预测，会话拦截，碎片，源路由，名字服务欺骗，欺骗路由包，欺骗控制包，端口扫描，伪装多目标和广播地址)。
- 抵抗攻击能力，搜集明显的恶意包的源地址或路由配置。
- IDS 产品或体系结构的容错能力。
- 在负载加重、冲突、电源故障、系统启动等不利环境下的 IDS 运行行为。
- 数据内容识别能力(病毒、可执行代码、Java Script，ActiveX 代码，邮件附件)。
- 冲突控制或流量管理的方法措施。
- 不同级别的报警和管理配置。
- IDS 通过何种方式报警(电子邮件，SNMP 触发器，送消息到控制台)。

6. **安全管理**

- 怎样灵活而又安全地控制访问 IDS。
- IDS 加密方式。
- 提供加密拨号连接管理。
- 管理员到控制台加密。
- 密钥交换协议和频繁更新密钥(适应 IPSec 协议，ISAKMP/Oakley 或 IKE)。
- 支持认证方式(Belleore S/key，Security Dynamics SeCUI-ID，Digital Pathways secureNet Key，CryptoCard RB-1，Enigma Logic)。
- 探测器和控制台之间的通信流是否加密。
- 按任务实施角色管理。
- 支持多种管理控制台。

- 自动完整性检查。
- IDS 产品怎样接入外部网，通过网络是否有办法可以访问 IDS 或攻击它。
- 根据带宽或积累的吞吐量测量监控到包比率或事件监控比率。
- 来自第三方的性能测试基准。
- 负载和网络带宽均衡。
- 过滤策略说明和实现是否容易。
- 过滤支持种类，包括协议、地址、服务、用户定义的模式。

7. 产品安装和服务支持

- 安装需求(处理器、RAM、硬盘)。
- 第三方编码规定要求。
- IDS 安装前所需软件(网络管理系统、操作系统、数据库系统)。
- 现有的路由器或主机是否变动。
- 软硬件安装是否容易。
- 默认设置(检测服务设置或去掉、日志、报警方式)。
- 因产品的某种安全问题，厂商是否提供快速修补。
- 产品升级计划，是定期还是随机。
- 入侵标签升级计划，是定期还是随机。
- 升级分发方式是磁带、磁盘还是在线。
- 升级的入侵标签是否采用加密或数字签名。
- 所需安装 IDS 探测器或主体的数目、位置、管理控制方式。
- 扩充需求。

4.8 入侵检测评估现状

美国政府 1998 年起开始入侵检测系统评估研究工作，评估在离线和实时两种情况下进行。根据网络公布的资料，下面简要地介绍这两种类型的评估方法。

4.8.1 离线评估方案

入侵检测系统离线评估是由美国国防部高级研究计划局与空军研究实验室联合发起的、由麻省理工学院林肯试验室主持进行的一年一度的评估活动，1998 年是第一次。此项活动将为入侵检测领域的研究指明努力方向并对现有系统的性能进行评估，因此具有深远的意义。其意图是引起所有从事主机或网络入侵检测工作的研究者的兴趣和关心。此项评估设计简单，集中于核心技术，消除安全与隐私问题并提供大多数入侵检测系统所使用的数据类型，鼓励厂商和研究部门参与。到目前为止，离线评估已开展了两次。

1. 技术目标

评估主要是测量入侵检测系统对发生于计算机系统或网络上的攻击的检测能力。本年度的任务集中于 UNIX 工作站，其目标是检测是否发生了下列攻击或在一个会话中存在下列的攻击企图：

- 拒绝服务(DOS)。
- 非授权的远程访问。
- 本地普通用户非授权使用超级用户特权或管理员权力。
- 监视探测。
- 用户的非授权访问、修改本地或远程主机的数据。
- 异常用户行为。

用计分方式表示网络会话是一个完整的 TCP/IP 连接,会话对应于多种服务,包括 telnet, HTTP,SMTP,FTP,finger,rlogin 等。其目的是与计算机和网络的正常使用环境对照。评估活动希望达到四个目标:

- 探索新的有前途的入侵检测思想。
- 开发包含这种新思想的先进技术。
- 对这项先进技术进行测量。
- 比较各种新技术和现有系统的性能。

以前进行的评估活动只注意到系统对入侵的检测概率,没有注意到系统的误报警概率。现在要求评估同时对入侵检测率和误报警率进行测量,方法是通过在正常的通信会话背景中加入攻击性会话。

2. 评估

入侵检测系统的性能是根据其对会话集进行检测的正确性进行度量的,这些会话集是通过模拟正常通信和攻击性通信产生的。正常的会话被设计成能反映出正常行为的通信量。攻击性会话包括新的攻击以及计算机非法使用所表现出来的特征,要求每一个入侵检测系统能对每一个会话打分,以反映其攻击的可能性。分值可用任何浮点形式(正、负或零)记录,并且按照惯例,正的分值越高,表示攻击的可能性越大。

对于任何给定的阈值 T,应该能够计算出系统对攻击的检测概率(分值高于 T 的攻击性会话的数量除以攻击性会话的总数量)以及误报警概率(分值高于 T 的正常会话的数量除以正常会话的总数量)。改变 T 的值可以绘出系统的操作特性 ROC 曲线,反映出攻击检测概率与误报警概率的对比。此操作特性曲线可用来确定系统在每个操作点的性能,以及不同的入侵检测方法之间的比较。操作特性曲线可由不同类型的攻击和异常行为产生,也可以通过将 BSM 数据输入系统或仅使用 TCPDUMP 转储或两者并用的方法产生。

3. 训练数据

在评估之前,每个参与的站点可以得到一批训练数据。这些数据可用来配置入侵检测系统或训练系统参数。一般来说,训练数据的类型与常见商用或科研用入侵检测系统使用的类型相同。这些数据由一个仿真网络产生,包含正常的和攻击性的会话。正常会话的类型和内容的分配与正常数据相似。攻击性会话将包括现有的攻击和计算机非法使用所表现出来的特性。训练数据包括以下内容:

1) TCPDUMP 转储数据,由 TCP 包嗅探器通过收集大约一个月的网络通信产生。该数据包括每一个在模拟的网络内外的计算机上转发的数据包。关于如何引用 TCPDUMP 转储的文档也将被提供。

2) 一个 TCP 转储数据的列表文件,每一个 TCP/IP 会话包含下列信息。

- 会话 ID:一个正整数。
- 开始日期:MM/DD/YYYY 格式。
- 开始时间:HH:MM:SS 格式。
- 会话持续时间:HH:MM:SS 格式。
- 服务标识:一个标识服务类型并指示是否 TCP 或 UDP 的字符串。如果一个服务以/u 结尾,则表明是一个 UDP 服务,否则认为是 TCP 服务。
- 源端口:一个正的整数,例如 1755,10500
- 目的端口:一个正的整数,例如 21,25。
- 源 IP 由点分开的四个非负整数,例如 192.168.1.30。
- 目的 IP 由点分开的四个非负整数,例如 192.168.1.31。
- 攻击分值:0 表示非攻击性会话,1 表示攻击性会话。
- 攻击名称:字符串((例如"guess","anomaly"等,"-"表示非攻击)。

列表文件以 ASCII 形式存在。各个域之间用白空格分开,不同的记录用换行符分开。列表文件仅包括 TCP 转储文件中的部分数据的信息。下面是一个 TCPDUMP 转储列表文件的示例:

```
11 01/23/1998 16:56:27 00:00:00 ftp-data 20 1770 192.168.020 192.168.1.30 0 -
13 01/23/1998 16:56:36 00:00:03 finger 1772 79 192.168.1.30 192.168.0.20 0 -
14 01/23/1998 16:56:42 00:00:03 smtp 1778 25 192.168.1.30 192.168.0.20 0 -
15 01/23/1998 16:56:43 00:00:03 smtp 1783 25 192.168.1.30 192.168.0.20 0 -
18 01/23/1998 16:56:45 00:00:00 http 1784 80 192.168.1.30 192.168.0.40 1 phf
20 01/23/1998 16:56:49 00:00:14 ftp 43504 21 192.168.0.40 192.168.1.30 0 -
```

3) Sun 工作站 BSM(Basic Security Module)提供的审计数据。此数据包括描述系统内核调用的审计信息。

4) BSM 数据的列表文件,其格式与 TCPDUMP 转储数据的列表文件相同。同样,仅有部分 BSM 捕获的网络会话被包含在此列表文件中。

5) 一个 ps-EFL 输出结果的文件,该文件内容是 UNIX 进程状态。

6) UNIX 转储数据,包含运行于 BSM 审计的机器的每个文件系统一星期或一天的增量转储。

7) 一个 PS 格式的图形文件,显示仿真网络中机器的逻辑组织及相互之间的路由器连接。

会话序列从 1 开始编号。一些会话仅出现在 TCPDUMP 数据中,另一些会话仅出现在审计数据中,还有一些在两者之中都出现。TCPDUMP 与审计数据之间的会话 ID 号是一致的。训练数据将以光盘的形式发布,预计大约有 10GB 大小。

4. 测试数据开发

测试数据开发用于在最后官方测试之前对入侵检测系统的性能进行评估。各个站点可以训练数据对系统进行训练,然后用预先指定的测试数据进行初步测试,并选择适当的系统设置使最后的测试表现良好的性能。使用统一的测试数据能够对不同的方法进行比较。

一般情况下,测试数据开发将按照与训练数据类似的方法产生。除了攻击分值和攻击名两个域为空之外,测试数据的各元素的格式均与训练数据相同。测试数据的格式为一个列表

文件,包含三个栏目:会话 ID、分值(0 为正常,1 为攻击)和攻击名。

以 1998 年的测试为例,将训练数据分成两部分:训练部分和开发用的测试部分。例如,如果训练数据包括了四周的数据,则最后一周的数据为开发用测试数据。建议各个站点用前三周的数据进行训练,用最后一周的数据进行测试。因为正确答案已包含在随训练数据一起提供的列表文件之中,故不再提供测试部分的答案。

5. 评估使用的测试数据

评估使用的测试数据或简单称为测试数据,是用来最后对各个入侵检测系统的性能进行测评的数据。评估用的测试数据将按照与训练数据及开发用的测试数据类似的方法产生。评估用的测试数据的各元素的格式均与开发用的测试数据相同,但正确答案要等到评估全部完成以后才能公布。在评估用的测试数据中会有在训练数据与开发用测试数据中不曾有的攻击类型。

6. 异常检测

一些入侵检测系统设计成能检测异常的用户、系统或网络行为。我们在训练数据与测试数据中将增设异常行为,以便对此类系统进行评估。在训练数据、开发用测试数据和最后测试数据中将保持用户、系统及网络行为的一致性。在三个数据集之间保持相同的用户和网络配置,以使在模仿正常的增删用户与服务时有很少的例外。另外,我们使测试数据在训练数据之后产生,以保持时间上连续性。这样,一个具有时间适应性的异常检测系统先用训练数据进行训练。

7. 评估规则

参试站点应提交三个正式的结果文件:第一个对应于 TCP 转储列表文件中的会话;第二个对应于 BSM 列表文件中的会话;第三个对应于二者会话的合并。尽管我们知道一些站点会只提交部分文件,但还是鼓励提交全部的三个结果文件。

允许在单一站点上进行多个系统的评测。比如一个站点可以为系统 A 提交三个结果文件,为系统 B 也提交三个结果文件。但在进行评估之前,必须指定一个系统作为主要受测系统。

提交的每个结果文件中的每个会话,都必须对其攻击可能性的大小进行打分。对于在列表文件中不能提供完整结果的参试站点,林肯试验室将不产生任何测试报告。也就是说,如果一个站点提供的 TCPDUMP 列表文件中包含 2000 个网络会话,那么也必须提供全部 2000 个会话的记分值。

所有参试者必须注意下列评估规则和约束条件。

- 每一个判断都必须以具体的、确已发生的网络会话为基础,不允许使用后来的会话信息。入侵检测系统必须是因果式的。
- 允许使用训练的条件信息(数据集目录结构隐藏的及其他网络信息提供的)。
- 在全部评估完成之前不允许检查评估数据,或其他与此数据的实验交互。本条规定适用于所有的评估测试数据,不管是部分的会话评估或者全部。

8. 结果提交格式

参加评估的所有站点必须提交所有会话的测试结果。提供给林肯试验室的结果必须用标准 ASCII 记录格式,每个判定形成一条记录。每条记录有三个域,彼此用空格分开。第一个

域是由林肯实验室赋予的会话标识符;第二个域是标识该会话具有的攻击可能性大小的浮点分值;第三个域(可选)是攻击名。不同的记录由换行符分开。结果文件必须在提交的最后期限前上传到林肯实验室的外部 FTP 站点。

9. 系统描述

系统描述要求参加评估的每一个系统,必须在提供结果文件时,将系统的名字与算法的简要描述一并提交。

10. 运行时间

参加测试的 IDS 系统必须提供相当于单个 CPU 对测试数据进行处理时所需的 CPU 时间。还必须给出 CPU 及所用内存大小的描述。

4.8.2 实时评估方案

作为由美国国防部高级研究计划局和空军 Rome 实验室(DARPA/AFRL)共同发起的 1998 年入侵检测系统评估计划的一部分,空军 Rome 实验室(AFRL)负责对选送的 DARPA 研究项目进行实时评估。其主要目的是为发起者和研究者提供一个度量标准,以判别一个研究项目的实际运行情况,并且为国防部选送在系统集成或进一步开发中比较有前途的技术。比较有前途的技术将被选入由 DISA 发起的 IA:AIDE 工程(Information Assurance: Automated Intrusion Detection Environment),这个工程用于在整个军用方面入侵检测技术的示范。林肯实验室的评估是用众所周知的传感器对系统的各个组成部分进行严格的测试,作为对它的补充,AFRL 主要对能作为现行网络中的一部分的完整系统进行测试。每个入侵检测系统允许使用所有的传感器(比如,活动检测以及可加载的软件模块等)并能产生一系列的响应。另外,AFRL 希望能对在非实时情况下很难检测的某些特征,如反应时间和重荷下的性能,进行测试。

1. 目标

评估有两个目标:

- 测量每个 IDS 系统在现有的正常机器和网络活动中检测入侵行为的效力。
- 测量每个 IDS 系统的反应机制的效力以及对正常用户的影响。

与 Lincoln 实验室相似,主要用下列攻击方式对受测系统进行测试:

- 拒绝服务(DOS)。
- 非授权的远程访问。
- 本地普通用户非授权使用超级用户特权或管理员权力。
- 监视探测。
- 用户的非授权访问、修改本地或远程主机的数据。
- 异常用户行为。

很多在 Lincoln 实验室使用的方法会被重复使用。除了上面列举的攻击方式以外,AFRL 测试还包括对网络基础结构进行的攻击和只有使用多级入侵检测系统才能检测到的攻击。还有离线测试中未使用的对操作系统进行的攻击。

对于每个受测系统将使用同一方案进行测试,但并不要求每个受测系统捕获每个攻击,也不产生能用于对受测系统的业绩进行排序的数据。本测试只对受测系统测试方案的覆盖程度

和特点进行评估。

2. 测试配置

每一次评估测试都在可控制的环境下进行。测试所用的网络与其他网络相隔离。控制软件按相同的次序对每个受测系统发出正常或攻击活动。自然，并不保证每次运行的数据包都精确一致，因为在这样一个复杂的系统下，一个小的变化都可能引起事件发生时间和顺序的变化。由于这些变化可能会潜在地引起运行过程中不同的误报警次数，要采取一些措施以消除这种不良效果。

在测试网络中，正常的通信由预期的脚本来产生，并且与 Lincoln 实验室中采用的会话相同。网络中会有各种各样的服务形式存在，并且在各个区域都可能产生连接或断开连接。入侵通信通过攻击实施方法来产生，这些攻击方法包括已知的攻击和新近出现的攻击方法。攻击可能来自网络内部也可能来自网络外部。

3. 测试要求

测试要求为了在不同的系统之间进行比较，制定一些标准，以利于评估的一致性。规定如下要求：

(1) 集成。为保证平稳地集成每个系统，研制者应仔细检查那些相关文档信息，在评估前提供下列信息：

- 所有需要安装软件的主机。
- 安装程序简要的描述。
- 所需的第三方软件。
- 所需安装的硬件。

首先，应该提供给 AFRL 一个功能性的版本，以便能够进行安装演示。在进行集成期间，研制者可以自由参观 AFRL 以确认系统安装正确并能按期望运行。

(2) 送交受测系统。受测系统应该按时间表上确定的时间送交 AFRL，之后不能做任何修改。

(3) 报告结果。Lincoln 实验室进行的离线测试是将会话清楚地列在一个文件中。与此不同，这里要求受测系统能在报告中辨认每次会话的连接数据以及时标。这样可测量系统的反应时间。与林肯实验室的测试类似，应对每个会话进行记分。默认时，如果 IDS 系统的一次会话没有分值，则按系统受测过程中报告的最低分计算：如果一次会话有多个分值，则按其最高分计算。输出的报告应具有下面的格式：{报告的时标}{会话开始时间}{持续时间}{服务器}{客户端口}{服务器 IP}{客户端 IP}{分值}。

4.9　练习题

一、选择题

1. 影响入侵检测性能的参数主要有(　　)。

A. 检测率　　B. 信息可信度

C. 虚警率　　D. 分析效率

2. 误用检测失效的原因有(　　)。

A. 系统活动记录未能为 IDS 提供足够的信息用来检测入侵

B. 入侵签名数据库中没有某种入侵攻击签名

C. 模式匹配算法不能从系统活动记录中识别出入侵签名

D. 入侵攻击签名不详细。

3. 性能测试与功能测试有所不同,因而测试的网络环境以(　　)为主,这样可以尽最大可能的产生大的网络流量,避免路由器等设备对网络流量的约束。

A. 虚拟网　　B. 局域网

C. 城域网　　D. 广域网

4. (　　)是测试软件的一个必不可少的功能,通过运行攻击来验证 IDS 是否能够检测到这些攻击。

A. 模拟攻击　　B. 实时攻击

C. 评估　　D. 攻击签名

5. 当购买一个管理器、代理结构的 IDS 应用程序时,主要考虑的因素是(　　)。

A. 管理器和代理之间的加密　　B. 代理穿越防火墙的能力

C. 管理器的位置　　D. 代理的位置

6. 当发现原有的 IDS 不能检测到新的攻击类型时,应采取的措施是(　　)。

A. 购买或更新特征库

B. 配置防火墙

C. 关闭 IDS 直到得到新的 IDS 应用程序

D. 定义一个新的规则来检测攻击

7. 如果要安装一个基于网络的 IDS 系统,首要考虑的是(　　)。

A. 网卡的性能　　B. 系统的供应商

C. 显示器的质量　　D. 内存的性能

8. (　　)属于评价 IDS 的性能指标。

A. 准确性　　B. 处理性能

C. 完备性　　D. 以上全部

9. (　　)属于 NIDS 的缺点。

A. 协同工作能力弱　　B. 很难检测复杂的需要大量计算的攻击

C. 难以处理加密的会话　　D. 以上全部

二、简答题

1. 简述检测系统的报警信息可信度与虚警率、检测率之间的关系。

2. 简述评价入侵检测系统性能的三个因素及其所表示的含义。

3. 简述 IDS 测试方法的局限性。

4. 简述性能测试的主要指标。

5. 简述离线评估方案和实时评估方案各自的优缺点。

第5章　主要入侵检测系统分析及入侵检测的标准化工作

本章导读：

本章首先介绍现有的主要入侵检测系统，然后对入侵检测技术的标准化工作进行介绍，虽然标准化工作目前还不是很有成效，但是标准化毕竟是方向。

5.1　国外主要入侵检测系统简介

5.1.1　RealSecure

实时入侵检测系统 RealSecure 是第一个集成了基于网络和基于主机的入侵检测商业产品。它可以最大限度地、全天候地监控企业级的安全问题。安全管理人员可以自动地监控网络的数据流、主机的日志等，对可疑的事件给予检测和响应，在内联网和外联网的主机和网络遭受破坏之前阻止非法的入侵行为。

实时入侵检测系统 RealSecure 包含三个部件：

- RealSecure Network Sensor 网络传感器(基于网络的监控器)
- RealSecure OS Sensor 操作系统传感器(基于主机的监控器)
- RealSecure Workgroup Manager 工作组管理器（管理控制台）

1. 网络传感器

RealSecure Network Sensor 运行在专门的工作站上，对网络入侵进行检测和响应。每一个 Network Sensor 监控一个网段，通过对网络上流过的数据包进行攻击特征的检测，识别出非法的企图并给予响应。

Network Sensor 一旦检测到非法行为就立即做出响应，响应的方式包括：中断非法连接，发出电子邮件或传呼机警告，记录非法事件过程和时间，重新配置防火墙或其他网络设备，执行用户指定的操作。另外，Network Sensor 还可以向 RealSecure Workgroup Manager 或第三方管理控制台发出警报，由管理人员进行后续的处理和检查。

2. 操作系统传感器

RealSecure OS Sensor 是基于主机的防范系统，是对 RealSecure Network Sensor 的补充。OS Sensor 分析主机的日志等内容来识别入侵和攻击，判断入侵是否成功，并且提供适合成为法律证据而实时条件又无法提供的信息。

每一个 OS Sensor 安装在工作站或主机上，彻底地检查系统日志，对非法入侵和网络误用进行匹配。OS Sensor 可以通过中断用户进程和禁止用户账户的方式防范入侵行为。它还可以发出警告，记录事件，发出 SNMP Trap，发电子邮件，或者运行用户定义的其他动作。

3. 工作组管理器

RealSecure Workgroup Manager 是 RealSecure 系统的控制台。可以对多台网络传感器和操作系统传感器进行管理。对被管理传感器进行远程的配置和控制，各个监控器发现的安全事件都实时地报告控制台。

4. RealSecure 的入侵发现过程

RealSecure 监控器是落实和强制执行机构基本安全策略的有效手段。一个 RealSecure 监控器的基本结构可以看作一个一般过程模块。

过程的输入包括：

- 用户或者安全管理人员定义的配置规则。
- 作为事件检测的原始数据。对于 Network Network Sensor 来讲原始数据是原始网络包；对于 OS Sensor 来讲原始数据是操作系统日志。

过程的输出包括：系统发起的对安全事件的响应。

过程如下：监控器在收到数据后，将数据和特征库进行对比，如果匹配，会发出对应的响应。特征数据库主要来源于 ISS 的 Xforce 研究开发小组，而且是目前商业领域最全面的攻击特征数据库。另外，Network Sensor 和 OS Sensor 都支持用户自定义特征。

5. RealSecure 的特征和响应配置

首先，当一个攻击或事件被检测到，RealSecure 可以做出以下几种响应：

- 自动终止攻击。
- 终止用户连接。
- 禁止用户账号。
- 重新配置 Check Point Firewall-1 防火墙阻塞攻击的源地址。
- 向 Lucent 防火墙安全管理服务器(SMS)发出实时警报。
- 向管理控制台发出警告指出事件的发生。
- 向网络管理平台发出 SNMP trap。
- 记录事件的日志，包括日期、时间、源地址、目的地址、描述以及事件相关的原始数据。
- 实时观看事件中的原始记录(或者记录下来过后再回放)。
- 向安全管理人员发出提示性的电子邮件。
- 执行一个用户自定义程序。

其次，可以为 RealSecure OS Sensor 自定义一些特征。OS Sensor 允许用户指定一个关键字或正则表达式，在发现操作系统日志文件项对应特征时，会做出相应的响应。

第三，用户可以为 RealSecure network Sensor 自定义连接事件。一个连接事件可以定义为基于 IP 的连接，可以对下面信息进行匹配：

- 协议。
- 源 IP 地址。
- 目的 IP 地址。
- 源端口。
- 目的端口。

第四，一些 RealSecure 预定义攻击特征可以根据网络的具体情况进行参数调整。比如，有

些网络中 PointCast 下载会引发 SYN Flood 特征，此时可以通过调整 SYN Flood 的阈值来减少误报。

第五，用户可以定义 RealSecure network Sensor 过滤规则来忽略某些类型的数据流。可以通过指定协议、源 IP 地址、目的 IP 地址、源端口、目的端口定义忽略的数据流。对于规则匹配的数据流不进行预定义或用户定义特征分析。通过这种方式，可以将 RealSecure 配置得更适合用户的网络。

最后，用户可以建立自己的响应选项。任何可以从命令行发起的动作（如：可执行程序、批处理文件、Shell Script 等），都可以作为 RealSecure network Sensor 或者 OS Sensor 对攻击的响应。

5.1.2　Cisco 公司的 Cisco Secure IDS

Cisco Secure IDS 以前被称为 NetRanger。Cisco 公司在网络硬件设备技术领域处于领先位置，因此他的入侵检测产品和其生产的硬件设备有着一定的联系，这也是他的主要特征之一。

Cisco Secure IDS 由控制器（Director）、传感器（Sensor）和入侵检测系统模块 IDSM（Intrusion Detection System Module）三大组成部分组成。传感器分为网络传感器（NIDS）和主机传感器（HIDS）两种，分别负责对网络信息和主机信息的收集和分析处理。控制器用于对系统进行控制和管理。Cisco 的 IDSM 最具有自身特点，即和硬件相联系，比如针对 Cisco 公司的 Catalyst 6000 交换机有相应的 Catalyst 6000 IDS Module 入侵检测系统模块。

1. 网络传感器

NIDS 检测器安装在多个位置上。最重要的位置是防火墙前面，负责监控进入机构的通信信息。另外，每个重要的网段都安装一个检测器。

网络检测器能够为网络设备及服务器上的通信模块提供全面保护。其主要特性有：

- 积极响应：系统包含对检测器设备的主动响应功能，用户只需修改 Cisco 路由器上的访问控制表（ACL）就能让系统自动回避或取消特定连接。回避功能可以临时启用，也可以长久保留。其他网络流量正常流动，只快速、有效地删除来自内部用户或外部入侵者的非法流量。这样，安全操作员就能够快速终止误操作，并防止入侵者访问网络。
- 全面检测网络袭击：包括检测对路由器和交换机的恶意袭击，检测面向服务器通信模块的第 3 层（或更低层次）袭击，检测探听或映射等企图，例如通常作为实际袭击前兆的呼叫清除和端口清除。
- 全面检测应用袭击：系统支持多种应用协议，例如 HTTP、DNS、文件传输协议（FTP）及其他协议。另外，它还能检测针对易损 CGI 程序发起的多种通信袭击。
- 以独特的方式预防 DDoS 攻击：检测 DDoS 代理与黑客之间的通信，寻找知名的 DDoS 工具，例如 Trin00 和 Tribe Flood Network（TFN）。
- IP 分片重装和“Whisker”反 IDS 检测功能支持：许多产品已经能够预防用于穿越典型 NIDS 技术的分组分片及其他编码技巧技术，但效果都不如 Cisco IDS 网络检测器。

2. 主机传感器

HIDS 首先部署在面对互联网的服务器上，例如 Web、邮件和 DNS 服务器。由于面向互联网的服务器与后端服务器相连，因此，HIDS 也部署在公司防火墙内的所有其他主要服务器上。

主机检测器能够为服务器上运行的服务器操作系统和应用提供全面保护。主机检测器安装在每台服务器上,用于保护 OS 和应用。系统利用呼叫截获技术提供纯主动式服务器安全系统。其主要特性有:

- 现场预防 OS 和应用袭击:与只有在袭击成功之后才查看记录和反应的基于记录的 HIDS 不同,主机检测器能够在袭击发生之前在呼叫水平上预防袭击。
- 防止缓冲器溢出袭击:主机检测器能够发现注入代码的执行过程并防止系统受损。两个主要功能如下:保护与用于提供代码的手段无关,即使注入代码未通过线路传输,也能防止袭击;机制使用了即使袭击未知也能防止执行恶意代码的通用签名,对于厂商尚未提供任何补丁程序的未知缓冲器溢出,这种方法能够提供保护。
- 不断提高完整性:通过控制对二进制、配置数据及其他系统对象的访问,主机检测器能锁定系统。即使是超级用户,也无法篡改系统。通过配置,主机检测器可以不允许修改某些系统设置,以保证设置符合推荐的默认值。这些特性不但能强化服务器操作系统,还能显著提高系统的完整性。
- Web 服务器屏蔽:为保护领先的 Web 服务器(IIS、Apache 和 I-Planet),服务器检测器包括特殊屏蔽模块。这些模块基于行为模块,能提供两种主要特性:防止其他程序访问特殊应用资源(甚至在权限用户执行的时候);防止恶意使用 Web 服务器。借助 Web 服务器专用的行为模式,主机检测器能防止未知袭击,因为识别基于行为模式,且不是每次袭击都需要特殊签名。
- 防止安全套接层(SSL)加密的 HTTP 袭击:NIDS 不能对用 SSL 加密的 HTTP 请求进行解密。利用这个弱点,黑客可以通过对袭击加密绕过 NIDS。加密后的恶意请求能够悄悄通过 NIDS,然后由 Web 服务器解密并执行,从而成功地利用易损点。另一方面,主机检测器则与 Web 服务器相连,能够在解密后服务前立即截获请求。如果发现请求是恶意的,请求将被丢弃,而不会发送到 Web 服务器,这样就可以预防袭击。

3. IDSM

这里以 Cisco 最新的 IDSM-2 为例来介绍 Cisco 的入侵检测系统模块。IDSM-2 是 Cisco 入侵检测系统的组成部分。它可以与其他组件合作,有效地保护数据基础设施。

IDSM-2 可以用在广泛部署的 Cisco Catalyst 系列设备上。Cisco IDSM-2 可以提供下列特性和优点:

- Cisco 是惟一可以提供一个交换机内置 IDS 解决方案的厂商,这种解决方案可以通过 VLAN 访问控制列表(VACL)获取功能来提供对数据流的访问权限。VACL 可以支持无限个 VLAN。
- 通过被动的、综合的操作提供透明的操作。这种操作方式可以通过 VACL 获取功能和交换机端口分析工具/远程 SPAN(RSPAN/SPAN)检测分组的复本,而且如果设备需要维护,因为它并不位于交换机转发路径上,因而它不会导致网络性能降低或者中断。
- 体积只占一个机架单元,在 Cisco Catalyst 设备中只占用一个插槽,从而使它成为能够有效地支持所有 Catalyst 设备(从有三个插槽的 Catalyst 6503 到这个系列中最大的设备)的平台,并让用户可以根据自己的需要,同时安装多个模块,为更多的 VLAN 和流量提供保护。

- 检测能力:Mbit/s 的 IDS 检测能力可以提供高速的分组检查功能,让用户可以为各种类型的网络和流量提供更多的保护。
- 多种用于获取和响应的技术,包括 SPAN/RSPAN 和 VACL 获取功能,以及屏蔽和 TCP 重置功能,从而让用户可以监控不同的网段和流量,同时让产品可以采取及时的措施,以消除威胁。
- 使用与 Cisco IDS 网络设备相同的程序代码,让用户可以将单一的管理技术作为标准,并让安装、培训、操作和支持变得更加便捷,同时可以利用 Cisco IDS 的攻击识别能力和特征库。
- 重要的管理技术(例如 Cisco VMS 2.1 安全产品包提供的支持,以及内置的 Cisco IDS 设备管理器(IDM)、IDS 事件浏览器(IEV)本地管理功能和 CLI 支持)让 IDSM-2 更加便于管理,更加善于检测和响应威胁,同时就潜在的攻击向管理人员发出警报。此外,这个新的产品还让管理人员可以更加方便地在范围广泛、多样化的网络上管理多个设备。

IDSM-2 的性能指标如下:

- 对 450 字节的分组的处理能力为 600Mbit/s。
- 每秒最多可以支持 5000 个 TCP 连接(新到达)。
- 最多可以支持 50 万个并发连接。
- 100%警报率。
- 在 Cisco Catalyst 机箱中添加 VLAN 或者设备不会对 Catalyst 的性能造成任何影响。
- 支持矩阵。

5.1.3　AAFID

AAFID 的全称是 Autonomous Agents for Intrusion Detection,是由美国 Purdue 大学提出的。在 AAFID 系统中,Agent 用来收集信息并进行一些基本的处理和判断,然后向上传递,实际上起的是收集、过滤和判断的作用;Autonomous 的意思是自治,在 AAFID 中主要是强调 Agent 的独立性。

AAFID 最初提出的时候只是一个框架,是入侵检测系统设计的一种方法,而设计者们在提出了这个框架后,为了证明设计的有效性,实现了一个原型系统,也起名为 AAFID。AAFID 是用 Perl 语言编写的,支持 Linux 和 Solaris 操作系统。

AAFID 的精髓在于其 Agent 系统,每个 Agent 都独立运行,收集它所感兴趣的信息并向上汇报。它可以直接将数据上传,也可以对数据进行处理和判断,在认为有必要的时候,也就是认为自己发现了异常现象或攻击行为的时候再向上级报告。就 AAFID 的体系结构而言,它的代理可以是一个完整的入侵检测系统,但是,作为一个分布式的入侵检测系统,AAFID 的设计目标就是实现分布和尽量简单的 Agent,因此不会用一个完整的入侵检测系统来充当 Agent。

作为一个分布式的入侵检测框架,AAFID 跨越了基于主机的入侵检测系统和基于网络的入侵检测系统之间的界限。自治 Agent 的引入使它可以广泛地收集整个网络环境内的各种信息,包括主机上的各种日志文件、网络上的各种数据包等。通过设计新的 Agent,可以针对各种数据源进行分析。Agent 甚至可以是一个网管系统的管理器,可以收集各个网络设备的信息。

AAFID体系结构主要包括四个部件:监视器、收发器、Agent 和过滤器。这些部件称为AAFID的实体。一个 AAFID 系统可以分布在网络中任意数目的主机上,在一台主机上可以运行任意数目的 Agent,Agent 可以通过过滤器来获取数据,也可以直接获取。每一台主机上的各个 Agent 将他们发现的事件和相关的数据传输给收发器。每台主机上只有一个收发器在运行,它监管本机所有 Agent 的运行情况,包括启动、停止 Agent,可以给 Agent 传递命令,也可以对从 Agent 收到的数据进行处理。收发器将自身汇总的数据报告给一个或多个监视器。每台监视器管理着多个收发器的运行,监视器可以看到整个网络范围内的数据,因此它可以进行高层次的相关性检查,进而检测到与多台主机相关的入侵。也可以将多台监视器按照层次进行组织,即有些监视器会向上一级的监视器进行汇报,或一个收发器向多个监视器进行汇报,这样可以提供数据冗余,避免单点故障。但无论层次如何划分,最终都必须有一个监视器使用用户界面来给最终的用户提供信息,通过这个界面给用户提供一个控制接口。

在 AAFID 系统中,实体基本上可以理解为一个程序,它独立运行并完成某种功能,AAFID 中的实体类型主要有:

- Agent:是一个独立运行的实体,它监视对应主机某个方面的运行情况,生成报告并向适当的收发器进行汇报。Agent 不产生任何警报,Agent 之间不进行通信,它们只能与收发器进行通信。在 AAFID 体系中,没有对 Agent 的功能进行界定,Agent 可以很简单,也可以很复杂。只要能按照 AAFID 的规定产生数据并进行汇报,它就可以作为 AAFID 系统的一部分来运行。
- 过滤器:为 Agent 提供数据的实体。它可以直接对数据进行提取,也可以进行选取,甚至加工。一个过滤器可以为多个 Agent 服务。
- 收发器:是每台主机对外通信的接口。它控制 Agent 的运行,并负责从 Agent 收集数据。在 AAFID 系统中,每台主机上必须运行一个收发器。
- 监视器:是 AAFID 系统中最高层的实体。具有与收发器类似的控制与数据处理功能,但监视器可以对多台主机上运行的实体进行控制,而收发器则不能。因为监视器可以从多台主机获取数据,因此它可以进行更高层次的相关性分析,还可以检查到收发器所无法检查的事件。监视器可以与用户界面进行通信,提供数据和控制机制。
- 用户界面:用户的最终接口,它与监视器进行通信、获取信息、发布命令。

概括来说,AAFID 系统的数据来源是 Agent 与过滤器,这两个实体对数据进行提取,并进行简单、低层次的处理。对于一个 AAFID 系统来说,检测类型的增加、检测功能的增强,都需要通过增加 Agent 和过滤器或者对现有的 Agent 和过滤器进行修改。因此,对于 AAFID 系统的用户而言,关键就在于根据自己的需要来编写新的 Agent 与过滤器。

5.2 国内主要入侵检测系统简介

5.2.1 “天眼”网络入侵检测系统

“天眼”网络入侵检测系统是由北京中科网威信息技术有限公司开发的基于网络的实时入侵检测及响应系统。该系统在产品的检测能力、响应能力以及系统自身的保护能力等方面都

进行了精心的设计，能够监视 10M/100M/1000M 局域以太网上传输的所有网络数据信息，根据用户指定的保护目标及检测策略对网络上传输的数据进行深度分析，当可疑行为或攻击行为发生时立即产生报警，同时根据用户需要能采取多种响应措施，包括立即切断连接会话、重新配置防火墙、发送 SNMP Trap 消息、发送电子邮件等。

系统采用引擎/控制台结构，网络引擎部署于网络中的各个关键点，通过网络和中央控制台交换信息。网络引擎负责网络数据的数据获取、分析、检测，对警报进行过滤和实时响应，并发送给控制台进行显示和记录；控制台负责警报信息的实时显示、记录、查阅等，并支持用户定制检测、响应策略和控制网络引擎。

1. 主要功能

"天眼"网络入侵检测系统具备一般网络入侵检测系统的主要功能，同时针对网络入侵检测系统面临的威胁和网络入侵检测系统发展方向开发出多项具有针对性的新功能，此外该系统对于国内外流行的防火墙提供互动接口，具有较完备的检测、响应、互动能力。其具体功能如下：

(1) 引擎集中管理功能

管理员在中央控制台可以直接控制各个引擎的行为，包括启动、停止、添加、删除引擎，也可以按照引擎查看、删除、查询实时警报。此外，在必要的时候用户还可以通过串口连接引擎主机，通过专用界面对引擎进行控制，包括启动、停止、查看/设置网络接口状态、查看引擎运行状态以及系统维护等。

(2) 策略管理功能

策略管理功能为用户提供了一个根据不同网段的危险程度，灵活配置安全策略的工具。网络管理员可以利用策略管理功能，轻松地针对特定的引擎定制策略，以满足不同网段对网络安全的需求，同时可以自定义规则。网络管理员还可以通过"对象管理"菜单，方便地对已经定制的策略进行网络对象、服务对象、响应方式以及响应对象修改。同时，策略管理功能还向用户提供了入侵检测规则的扩充能力，网络管理员可以根据自己的需要定制入侵检测规则，直接应用于网络引擎，很快实现网络管理员的安全意图。

(3) 实时入侵检测功能

"天眼"网络入侵检测系统能实时识别各种基于网络的攻击及其变形，包括 DOS 攻击、CGI 攻击、缓冲区溢出攻击、后门探测和活动等。该系统通过深入应用协议分析去除干扰并还原攻击本来面目，降低了误报率和漏报率。目前可以检测 26 大类、1000 余种攻击行为及其变形。

(4) 警报过滤功能

"天眼"网络入侵检测系统能根据定制的条件，过滤重复警报事件，减轻传输与响应的压力，同时还能保证警报信息不被遗漏。它能够明显缓解目前针对网络入侵检测系统的 DOS 攻击，使得系统能够保持不间断地稳定工作。

(5) 实时响应功能

根据用户定义，警报事件在经过系统过滤后进行及时响应，包括实时切断连接会话、重新配置防火墙、彻底屏蔽攻击、给管理员发送电子邮件、发送 SNMP　Trap 报警、控制台实时显示、数据库记录等。

(6) 报表统计和数据库维护功能

“天眼”网络入侵检测系统提供了非常简便的入侵警报统计和报表工具。用户可以根据各种可选条件，例如：引发报警数据包的源 IP 地址、目的 IP 地址、源端口号、目的端口号、警报产生的时间、危险级别等等，使用单一条件或复合条件进行查询，当警报信息数量大、信息来源广泛的时候，网络管理员可以很轻松地对警报信息进行分类，从而突出显示网络管理员需要的信息。

控制台程序长时间工作后会产生大量的冗余信息，通过压缩数据库，可以使数据库变得更精简，使程序工作效率更高、更稳定。当数据库达到一定大小的时候，通过“天眼”网络入侵检测系统的历史数据维护程序，网络管理员可以根据自己的需要导出某一个确切时间范围内的数据，进行备份，以保证程序的正常运行和日后的查看。数据导入功能则可以将备份的数据导入程序数据库，来查看历史警报信息。

(7) 强大的策略模板管理功能

系统默认提供六种模板，并且支持用户自定义模板，允许用户将自己定义的模板和系统的模板进行导入导出的操作。

(8) 提供大型数据库转换支持

控制台允许用户将警报的信息改用大型数据库来存取、管理，如 MS SQL Server 等。

(9) 策略库的在线升级支持

系统支持在线远程升级策略库，使系统的策略库时刻保持防范的最前沿。

2. 主要特点

“天眼”网络入侵检测系统具有如下主要特点：

(1) 引擎稳定高效

网络引擎软件由三个部分组成：引擎管理、数据获取与检测、警报过滤与响应，保证了网络数据处理和警报处理相对独立，不致阻塞；引擎管理代码逻辑简单，能保证长时间稳定运行，同时保证其他两部分稳定工作，确保网络引擎整体 24 小时不间断工作。各个数据处理单元间采用多级缓冲设计，确保高带宽下数据不致丢失。

(2) 支持灵活的系统部署

系统支持控制台同时作为客户端和服务器，即同时从一个网络引擎拉数据和向另外一个引擎推数据，这使得网络引擎和控制台的部署可以满足复杂网络拓扑的实际需求。对于隐藏本地 IP 地址的多网段大型网络环境，可以把控制台置于内部网络，而把网络引擎部署到网络中的任意位置。此外，在骨干网实际带宽很高的情况下，用户可以根据地址、协议、应用类型等条件配置数据获取策略，同时部署多个网络引擎确保充分检测网络数据，使得保护对象更加明确具体，提高引擎运行效率。

(3) 更低的误报率和漏报率

“天眼”网络入侵检测系统采用领先的基于应用的入侵检测技术，根据各种典型应用的具体协议对网络数据进行分析、检测，能够识别各种伪装或变形的攻击，降低了漏报率，同时去除了干扰，也降低了误报率。

在详细的协议分析的基础上，系统采用重组还原技术判断碎片攻击，对通过将 IP 包进行分片的方式来逃避 IDS 检测的攻击行为，“天眼”在每次匹配之前，先判断数据包是否为分片包，如果是，则根据协议规则先进行重组，使本次的请求还原到一个完整的数据包状态再进行检测。

常规的网络入侵检测系统一般只判断单一的数据包，因此对于一次可疑行为由两个或多个相关联的数据包共同来完成的攻击行为，就会产生漏报。“天眼”网络入侵检测系统会记录每一次会话的全过程，使那些上下文相关的入侵行为无处躲藏，从而使检测数据范围更广，检测更严谨。

此外，“天眼”网络入侵检测系统还在应用层针对特殊服务做了特殊编码还原/协议分析处理，使得能够检测到一些 IDS 逃避攻击、特殊编码变形等攻击行为。

(4) 增强的抗攻击能力

“天眼”网络入侵检测系统能够防御针对入侵检测系统的拒绝服务攻击，采用的重复事件过滤技术和其他监控手段，使得对入侵检测系统的拒绝服务攻击的冲击降到最低。同时对于网络中断等异常情况采用了智能控制技术，确保引擎稳定运行，警报不致丢失，当网络恢复后能够及时显示。

(5) 系统自身安全性更强

网络引擎运行于安全操作系统并经过严格配置。网络引擎具备管理微内核，采用双网卡工作方式，通信支持认证和加密，确保系统的安全。

(6) 千兆入侵检测

随着宽带网络的普及应用，千兆环境下的入侵检测已经日益受到重视。“天眼”网络入侵检测系统能够运行于千兆网络环境，基本能够满足千兆环境下的流量要求。

5.2.2 “天阗”黑客入侵检测系统

“天阗”黑客入侵检测与预警系统是启明星辰信息技术有限公司自行研制开发的入侵检测系统。“天阗”黑客入侵检测与预警系统是一种动态的入侵检测与响应系统。它能够实时监控网络传输，自动检测可疑行为，及时发现来自网络外部或内部的攻击，并可以实时响应，切断攻击方的连接。天阗系统可以与防火墙紧密结合，弥补了防火墙的访问控制不严密的问题。其联机文档提供了丰富的事件说明及恢复措施，可以最大程度地为网络系统提供安全保障。

针对不同的用户需求和应用环境，启明星辰公司推出了天阗入侵检测系统系列产品，包括有：

- S100：应用于百兆网络，主要应用于网络入侵检测。
- S500：应用于百兆网络，支持多级分布部署、可与主机版 IDS 混合管理和关联分析。
- S1100：应用于千兆网络，主要应用于高速网络入侵检测。
- S1500：应用于千兆网络，支持多级分布部署、可与主机版 IDS 混合管理和关联分析。
- S110：应用于 Windows 平台，为独立的主机入侵检测系统。
- S220：应用于 SUN 平台，为独立的主机入侵检测系统。
- S320：应用于 IBM 平台，为独立的主机入侵检测系统。

1. 网络型入侵检测系统

网络型入侵检测系统主要用于实时监控网络关键路径的信息，如同大楼内无处不在的闭路电视录像监控系统，它采用旁路方式全面侦听网上信息流，动态监视网络上流过的所有数据包，通过检测和实时分析，及时甚至提前发现非法或异常行为，并进行响应。通过采取告警、阻断和在线帮助等事件响应方式，以最快的速度阻止入侵事件的发生。网络型入侵检测系统能

够全天候进行日志记录和管理，进行离线分析，对特殊事件进行智能判断和回放。

“天阗”网络型入侵检测系统根据其检测功能和支持的网络带宽，划分为如下型号：

- N100：应用于百兆网络环境，实现基本的网络入侵检测功能。
- N500：应用于百兆网络环境，实现扩展型网络入侵检测，支持特征自定义、双百兆网段检测。
- N1100：应用于千兆网络环境，实现基本的高速网络入侵检测功能。
- N1500：应用于千兆网络环境，实现高速扩展型网络入侵检测，支持特征自定义、双百兆网段检测。

网络型入侵检测的主要优点如下：

- 按网段检测，方便实施和部署。
- 旁路方式监听，对入侵者和业务网络透明存在。
- 不占用网络带宽资源和其他业务主机的系统资源。
- 可以实现和其他安全设备之间的紧密配合。

2. 主机型入侵检测

主机型入侵检测系统是基于对主机系统信息和针对该主机的网络访问进行监测，及时发现外来入侵和系统级用户的非法操作行为。它可以用来实时监视可疑的连接、系统日志检查、非法访问的闯入等，并且提供对典型应用的监视，如 Web 服务器、邮件服务器等。

“天阗”主机型入侵检测系统根据主机平台和操作系统类型，划分为如下三种：

- H120：应用于 Windows 平台，可以为 Windows NT/2000、Windows XP、Windows 2003 等。
- H220：应用于 SUN 平台，要求为 Solaris 8 以上版本。
- H320：应用于 IBM 平台，要求为 AIX 4.3 以上版本。

主机型入侵检测的主要优点如下：

- 不受交换环境影响。
- 不受加密环境影响。
- 监测系统内部入侵和违规操作。
- 可以对本机进行防护和阻断。

3. 天阗入侵检测系统的特性

天阗入侵检测系统具有如下特性：

- 积极主动地对攻击做出响应，使得系统能够在恶意行为造成危害之前阻断他们。这样可以防止系统免受已知和未知攻击的侵害，最大限度地确保关键业务的正常运行和可用性。
- 预配置的策略集，包括完善的客户化选项。这样就减少了对于专职安全人员的需求时间。
- 具有防止恶意访问系统资源的能力。这同样减少了对于专职安全人员的需求时间。
- 易于使用。引擎的安装、配置直至开始工作的时间不超过 15 分钟。
- 无需专人持续监视。系统具有灵活多样的安全事件告警方式，可进行电子邮件和短消息等方式告警。
- 分级管理。可以对大规模网络进行有效安全监视和管理。
- 升级方便。可以快速、方便地进行维护，减少了管理人员的工作量。

4. 天阗网络入侵检测系统的组成

基于网络的入侵检测系统则主要用于实时监控网络关键路径的信息，在检测到入侵信息后，网络探测引擎向控制中心的管理控制台发出告警，由控制中心给出定位显示，帮助管理员将入侵者从网络中清除出去。

“天阗”网络型入侵检测与预警系统（简称天阗 NIDS）主要包括两部分组件：网络探测引擎和控制中心。网络探测引擎采用专用硬件设备通过旁路方式接入检测网络；控制中心面向用户，提供显示和管理配置之用。

网络探测引擎全面侦听网上信息流，动态监视网络上流过的所有数据包，进行检测和实时分析，从而实时甚至提前发现非法或异常行为，并且执行告警、阻断等功能并记录相应的事件日志。

控制中心是个高性能的管理系统，它能控制位于本地或远程的多个网络探测引擎的活动，集中制定和配置策略，提供统一的数据管理和实时告警管理。它能显示详细的入侵告警信息（如入侵者的 IP 地址、攻击特征），对事件的响应提供在线帮助，以最快的方式阻止入侵事件的发生。另外，它还能全面的接收和管理日志，以便进行事后分析并形成综合报告。

控制中心可以被设置为主控制中心或子控制中心。主控制中心除了具有控制中心的所有功能外，还可实时接收子控制中心的告警信息，定时、分类提取子控制中心的日志信息，下发各种配置文件、策略供子控制中心对其所属网络探测引擎进行配置。

5. 天阗的入侵检测能力

入侵检测能力取决于两个方面的技术：攻击特征分析技术和入侵检测支撑技术。天阗的攻击特征分析和研究完全由启明星辰自主承担，长期的积累形成了全面的攻击事件特征库。对于每一种攻击特征都做了严格的分析和验证检测，确保攻击特征的准确性，并建立和相关的网络入侵、主机入侵和漏洞之间的关联关系；对于新的入侵方式，启明星辰第一时间进行分析，并及时发布升级包到网站上供用户下载。同时，天阗攻击特征库可以通过灵活方便的语言由用户根据自身网络应用的特点进行自定义扩充，从而实现天阗的客户化应用检测。

在入侵检测的支撑技术上，天阗的技术优势体现在高速捕包、深入协议分析技术、高速树型匹配技术、防躲避处理技术以及事件风暴处理技术等多个方面，有效地保证了入侵检测的准确性、有效性和高性能。

- 引擎高速捕包技术。采用专门设计的以太网卡驱动程序和应用程序进程共享一块抓包数据缓存区。因为减少了内存拷贝的开销和上下文切换次数，零拷贝的接口能够提供更大的吞吐率，提高了报文捕获效率，同时 CPU 占用率大大降低。
- 深入协议分析技术。天阗采用协议树方式规范协议分析的流程，保证协议解析的高速度，产生相关的协议分析事件；对具体的协议分析深入到应用数据，对关键字段进行分组，同时支持搜索深度，为特征匹配提供准确定位，减少匹配的代价。
- 高速树型匹配技术。天阗采用的高速树型匹配技术实现了一次匹配多个规则的模式，在规则数目增多的情况下不会增加特征匹配需要消耗的时间和资源，因此，检测效率得以成倍的提高。
- 防躲避处理技术。天阗采用了 IP 碎片重组、TCP 流重组以及特殊应用编码解析等多种方式，应对躲避 IDS 检测的手法，如：WHISKER、FRAGROUTE 等攻击方式。

● 事件风暴处理技术。天阗内置了状态检测机制和基于事件的统计技术，可以识别和处理类似“STICK”等的反 IDS 攻击，并对同一事件采用合并上报，有效地避免了事件风暴的产生。

5.2.3 “冰之眼”网络入侵检测系统

“冰之眼”网络入侵检测系统是绿盟科技开发的网络入侵检测产品。“冰之眼”网络入侵检测系统由网络探测器、内网探测器、SSL 加密传输通道、中央控制台、日志分析系统、SQL 日志数据库系统及绿盟科技中央升级站点几部分组成。

1. 检测能力

入侵检测系统最核心的要求就是能够准确地对入侵行为进行识别，并采取有效措施进行防护和干预。但是由于技术原因，以及欺骗性攻击技术的不断发展，漏报和误报一直是困扰入侵检测领域的两大难题。绿盟科技专业的安全专家，在对各种攻击手段进行充分的研究后，总结了一套行之有效的检测手段，误报率、漏报率均远远低于国内外同类产品，并得到了权威机构的测评认可。

- 检测超过 1000 种的攻击方式，特别包括众多面对国内系统特有漏洞的攻击，这是国外产品所无法做到的。基于绿盟科技强大的研究实力，保证提取攻击特征的有效性，最大程度地降低漏报和误报。
- 优异的运行性能：百兆及千兆产品分别能够很好地满足于百兆及千兆网络，可以对高流量的数据进行有效报文采集和处理，不会因流量问题造成采集缺失而漏报。
- 利用碎片穿透技术突破防火墙和欺骗入侵检测系统已经成为黑客常用的手段，“冰之眼”网络入侵检测系统能够针对各种协议有效进行 IP 层的碎片重组，让碎片欺骗技术无所遁形。
- TCP 状态跟踪及流汇聚通过对 TCP 状态的检测，能够有效避免因单纯包匹配造成的误报，对于利用 Stick、Snort 工具进行欺骗攻击的 IDS Flood 行为能够有很好的甄别防范能力。
- 针对常用协议进行解码，能够“认清”数据的真实面目并提高了检测效率和准确率。
- 采用多种方式，对 SYN Flood、Winnuke、Jolt、Teardrop 等拒绝服务攻击进行检测，配合其他安全产品，能够进行有效的防御。

此外，“冰之眼”网络入侵检测系统加强了对内网行为（内网拨号、IP-MAC 对应关系改变、主机是否在线等发生在企业内网中的特定事件）的有效监控和跟踪，能够很好的帮助网络管理者发现和跟踪内网异常，确保对内网安全事件进行有效的监控管理。

2. 主动防御能力

传统的入侵检测系统仅仅是对入侵进行监控和报警，却没有很好的措施进行防御，因此在很多安全应用场合，入侵检测系统更多是提供事后亡羊补牢的参考，而不能在攻击进行时起到必要的防护作用，入侵检测系统的价值也就无法充分体现。绿盟科技认为：网络安全体系应该是通过完整的安全策略，利用多个安全产品的有效互动实施全方位保护，而不是单纯的安全产品的堆砌。基于这种认识，绿盟科技在“冰之眼”网络入侵检测系统产品中加强了与防火墙以及其他安全管理产品的联动功能，协助用户搭建真正可靠的安全防护体系。

通过对各种防火墙产品的联动操作，“冰之眼”网络入侵检测系统能够自动切断攻击行为，“冰之眼”网络入侵检测系统当前能够与多种不同种类的防火墙实现联动，真正起到主动防御作用。以下是“冰之眼”网络入侵检测系统支持联动的防火墙产品列表：

- 天融信防火墙。
- Check Point FW-1。
- 东方龙马防火墙。
- 天网(Sky Net)。

3. 管理能力

可管理性是入侵检测系统能够真正发挥作用的重要因素，一个入侵检测系统产品即便拥有好的技术，而缺乏有效的数据管理功能，依然无法让网络管理人员真正有效的利用起来，其价值也就无法得到体现。

绿盟科技做为国内最早从事专业安全服务的企业，将多年丰富的网络安全服务经验融入产品设计，确保用户能够方便地使用，从而最大可能地发挥入侵检测系统的防护效用，并真正协助用户实施安全决策和统筹安全管理。

以下是“冰之眼”网络入侵检测系统产品的安全管理特性：

- 具有信息过滤/合并功能，能够在控制台过滤一些低风险或者不可能成功的攻击行为，从而极大地减少管理员的工作量，也使得重要攻击行为能够得到重点体现。
- 会话记录及回放能够对指定来源或保护对象的 TCP 会话进行跟踪和回放，对于一些可能存在的危险行为，可以实施跟踪观察，方便管理员确定攻击者意图和攻击途径。
- 自定义检测规则，针对一些具有特殊安全防护的需求，比如内容信息过滤，可以方便设置自定义的关键字过滤检测规则，通过与防火墙的联动，也可以将该涉及敏感信息的 TCP 会话阻断，防止企业重要信息泄露或者一些非法的网络信息传递。
- 报表系统可以自动生成各种形式的攻击统计报表，包括日报表、月报表、年报表等，通过来源分析、目标分析、类别分析等多种分析方式，以直观、清晰的方式从总体上分析网络上发生的各种事件，为管理人员提供方便。
- 规则自动在线升级(同时也提供手动升级)、日志数据库的自动维护、报表的自动生成和发送等一整套自动化功能，有效地降低了管理员的工作强度，提高了产品持续运行的时间。
- 通过强大的集中监控中心实现分散信息采集和集中控制的思想。管理员在控制中心就能管理和综合地分析所有探测器的告警信息和状态信息，形成统一的分析报表。

4. 自身安全性

安全产品自身的安全性非常重要，否则将很难起到应有的保护作用，反而会成为网络安全体系的隐患。

- 防欺骗性攻击：一些别有用心的攻击者通过一些工具造成入侵检测系统大量误报，然后将入侵行为隐藏在大量无用的误报中。当精心构造的欺骗数据包达到一定规模，有可能导致入侵检测系统的崩溃。“冰之眼”网络入侵检测系统针对各种入侵检测系统欺骗方式和攻击方式进行了总结，并采用了基于状态的行为分析方式，有效避免了这种情况的发生，能够确保产品不会遭受诸如 Stick、Snort 的攻击。

- 多点备份功能:当网络出现故障时,探测器能够自动通过最多四条备用通道将数据发送到中央控制台,从最大程度上保证了告警消息的及时性和可靠性。
- SSL 加密传输通道:确保管理控制信息不会被恶意用户截获和窥探。

5.2.4 ERCIST-IDS 网络入侵检测系统

ERCIST-IDS 网络入侵检测系统是由中科安胜公司开发的入侵检测产品。它是以数据库为中心、基于客户/服务器模式的分布式网络入侵检测系统,它将软件划分成三层架构模式,由控制中心和嗅探器两大部分组成。安胜 IDS 的控制台操作界面采用 Windows 图形方式,用户可简单地操作控制安胜 IDS 的运行、监测嗅探器工作状态、配置入侵检测规则和报警综合信息分析。安胜入侵检测系统能够进行协议分析、内容的搜索/匹配。支持协议 TCP、UDP 和 ICMP 等协议分析。它能够检测多种方式的攻击,例如:缓冲区溢出、秘密端口扫描、CGI 攻击、SMB 探测、探测操作系统指纹特征的企图等。

1. **主要功能**

(1) 分布式网络嗅探

安胜 IDS 的嗅探器安装在寄居主机上或是以一个独立的黑匣子形式连接在网络的重要位置,用来检测网络上各种攻击行为,并将攻击报警信息发送到安全控制中心。

(2) 集中的安全控制与分析

安胜 IDS 的安全控制中心是管理嗅探器,包括规则配置和嗅探器运行状态监探。控制中心还对来自嗅探器的报警信息进行综合分析,给用户提供可视化报警结果。安全控制中心分为以下三大模块:报警监视器、嗅探控制器和嗅探器规则装配器。

(3) 实时入侵检测

拥有一个内容丰富、准确的入侵检测特征库使得安胜网络入侵检测系统可以检测很多种入侵行为,包括:拒绝服务攻击、溢出攻击、未授权访问、网络资源滥用、涉密信息传输、病毒传输等等。

(4) 动态特征库管理

在目前的版本中所收集的有 33 大类共 1633 条攻击监测规则,可以监测到目前可能遇到的几乎全部攻击行为。用户也可以通过互联网直接下载、更新入侵检测特征库。

(5) 全中文远程控制

网络入侵检测系统提供了独有的控制台界面,操作简单、容易掌握。所有信息全中文显示,例如警报信息、警报的详细描述等,简洁、亲切,便于使用。

(6) 嗅探数据的传输机密

传输加密技术的应用有效的保证了用户在移动管理过程中数据传输的安全性,包括用户名、用户密码等敏感信息均得到了有效的保护,进一步保证了整个入侵检测系统运行的安全性。

(7) 提供入侵响应功能

安胜 IDS 提供了多种入侵行为响应技术,用户可以根据网络的安全策略进行选择。包括:记录现场数据、发送邮件、阻断连接、分级上报等功能。分级上报又分为向上级数据库记录现场数据和向上级管理员发送警报邮件两种形式。

(8) 稳定、高效的网络嗅探器

网络引擎采用多线程设计,网络数据的采集、数据的分析处理、针对入侵行为的响应动作均独立进行,并在数据处理过程中进行了合理的分类,优化了处理过程,大大提高了网络引擎的工作效率,不仅达到了实时、准确、高效的基本要求,而且能够在较大数据流量的情况下保持稳定的性能和较小的丢包率。

(9) 跨平台技术的应用

安胜 IDS 可以满足不同用户选用不同操作系统的需求,在系统设计过程中采用了大量的跨平台技术和工具。目前版本不仅可以运行于 Window NT/2000 操作系统,而且也可以运行于 Linux 系统。

(10) 灵活、多变的应用方式

安胜 IDS 分为控制台和嗅探器两个完全独立的部分。网络管理员可以根据所需保护网络的具体情况和安全策略,灵活的布置网络引擎,实现对网络的立体式多层监探与保护。并且通过一个控制台就可以控制所有网络引擎,达到了分布式安装、集中管理,管理员可以高效的完成对大型网络的安全管理任务,提高工作效率。

(11) 实现大型网络的分层、分级管理

安胜 IDS 使用了严格的用户身份认证与控制机制,根据用户的权限赋予该用户对系统功能的使用能力。例如:规则定制与应用、警报结果查询与删除、用户的管理、网络引擎的管理、入侵行为的响应动作定制等等,使不同级别的网络管理员具备不同的对整个网络的管理能力。实现对大型网络的分层、分级管理。

(12) 提供多种入侵响应技术

安胜 IDS 提供了多种入侵行为响应技术,用户可以根据网络的安全策略进行选择。包括:记录现场数据、发送邮件、阻断连接、分级上报等功能。分级上报又分为向上级数据库记录现场数据和向上级管理员发送警报邮件两种形式。

(13) 轻量级的网络入侵检测系统

安胜 IDS 是一个轻量级的网络入侵检测系统(NIDS)。所谓的轻量级是指在检测时尽可能低地影响网络的正常操作,具备跨系统平台操作,对系统影响最小等特征并且管理员能够在短时间内通过修改配置进行实时的安全响应,更为重要的是能够成为整体安全结构的重要成员。

2. 主要特点

(1) 开放性:系统设计成以数据库为中心、基于客户/服务器模式的分布式入侵检测系统,易于扩充;对新的攻击威胁反应迅速。

(2) 可用性强:具有简单、易操作的图形化管理操作界面。

(3) 功能性强:将网络入侵检测和主机检测融为一个整体,便于入侵检测安全管理。能够检测常见的网络攻击。

(4) 适应广泛:安胜入侵检测系统的跨平台性能极好,支持 Linux、Window2000 等系统。

(5) 学习智能性:安胜入侵检测系统通过搜集网络数据和主机系统的操作信息,进行学习归纳,能够发现入侵行为。

(6) 专用的安全操作系统:嗅探器采用自主开发的安胜安全操作系统,安全可靠。

5.3 入侵检测的标准化工作

随着网络规模的扩大,网络入侵的方式、类型、特征各不相同,入侵的活动变得复杂而又难以捉摸。某些入侵的活动靠单一 IDS 不能检测出来,如分布式攻击。网络管理员常因缺少证据而无法追踪入侵者,入侵者仍然可以进行非法的活动。不同的 IDS 之间没有协作,结果造成缺少某种入侵模式而导致 IDS 不能发现新的入侵活动。目前,网络的安全也要求 IDS 能够与访问控制、应急、入侵追踪等系统交换信息,相互协作,形成一个整体有效的安全保障系统。然而,要达到这些要求,需要一个标准来加以指导,系统之间要有一个约定,如数据交换的格式、协作方式等。基于上述的因素考虑,国际上的一些研究组织正在开展这方面的研究工作。

总的来说,入侵检测技术在最近几年发展迅速,但是相应的入侵检测标准化工作则进展缓慢。当前有两个国际组织在进行这方面的工作,他们是 Common Intrusion Detection Framework(CIDF)和 Internet Engineering Task Force(IETF)下属的 Intrusion Detection Working Group(IDWG)。他们考虑了入侵检测的不同方面,并从各自的角度进行了标准化工作。下面介绍一下目前入侵检测标准化制定工作的进展状况。

5.3.1 CIDF 的标准化工作

入侵活动变得愈加广泛,而且许多攻击是经过长时期准备,通过网上协作进行的。面对这样的情况,入侵检测系统和它的构件之间共享这种类型的攻击信息是十分重要的。共享让系统之间对可能即将来临的攻击发出警报。为了实现这样的目标,IDS 系统应定义好共享信息的接口。为此,S.Staniford-Chen 等人提出了通用的入侵检测框架(CIDF)。

CIDF 标准化工作基于这样的思想:入侵行为是如此广泛和复杂,以至于依靠某个单一的 IDS 不可能检测出所有的入侵行为,因此需要一个 IDS 系统的合作来检测跨越网络或跨越较长时间段的不同的攻击。为了尽可能地减少标准化工作,CIDF 把 IDS 系统合作的重点放在了不同组件间的合作上。

CIDF 所做的主要工作有:提出了一个通用的入侵检测框架,然后进行这个框架中各个部件之间通信的协议和 API 的标准化,以达到不同 IDS 组件的通信和管理。当前 CIDF 包括以下四个方面的主要内容:

- Architecture:提出了 IDS 的通用体系结构,用以说明 IDS 各组件间通信的环境。
- Communication:说明 IDS 各种不同组件间如何通过网络进行通信。主要描述各组件间如何安全地建立连接以及安全地通信,包括组件间的鉴别和认证。
- Language:采用公共入侵规范语言 Common Intrusion Specification Language(CISL),IDS 各组件间通过 CISL 来进行入侵和警告等信息内容的通信。
- API:允许 IDS 各组件的重用,在 CISL 的表示说明中隐含了 API。

1. CIDF 的体系结构

CIDF 定义了 IDS 系统和应急系统之间通过交换数据,协作实现入侵检测和应急响应。CIDF 互操作有下面三类:

- 配置互操作,可相互发现并交换数据。

● 语法互操作，可正确识别交换的数据。

● 语义互操作，可相互正确理解交换的数据。

CIDF 定义了 IDS 系统的六种协同方式，介绍如下。

(1) 分析

如图 5-1 所示。A 搜集原始数据并可以作预处理分析。B 则根据 A 进行更深层次分析和产生报告。A 搜集原始事件记录器、信号分析或者异常检测器。B 作为后续的信号分析器，试图去断定由 A 报告的若干事件的相互关系。或者 B 作为异常检测器，尽力去判断来自 A 的报告的权重是否应引起注意。

(2) 互补

如图 5-2 所示。A1 和 A2 能够相互弥补。M 则负责合并 A1 和 A2 的输出结果。A1 和 A2 可以检测不同的各种各样的攻击。A1 可能检测到对 TCP/IP 协议的攻击，而 A2 则检测到对某个应用的攻击。或者 A1 和 A2 在不同的计算机上检测到同一种攻击。实际上，A1 和 A2 有可能是同一个检测程序或者不是一样的程序。这种情形用于推断或者趋势分析。若多个 A 开始报告一个特定的异常事件，M 就能检测出更多的异常入侵。假如 A 所报告的不准确，只是相互之间的报告重叠，那么 M 可能会做额外的工作。

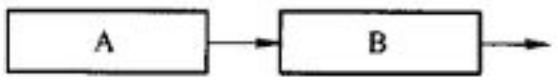

图 5-1　CIDF 协同方式：分析

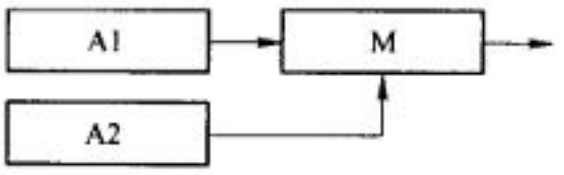

图 5-2　CIDF 协同方式：互补

(3) 互纠

如图 5-3 所示。A1 和 A2 两个构件可互相纠正检测结果。由于入侵检测中存在许多报警，故构件之间的互纠是必要的。但是，采用不同的分析方法两个检测器常会产生多次误报警，特殊情况是，J 构件在 A1 和 A2 的检测结果都相同的情况下才会报告入侵。例如，A1 可能作统计异常检测，只有在网络连接数超过一个固定的阈值后才会报告入侵。而 A2 是根据入侵标签来检测，当 A2 观测到已知的攻击模式时才会报告入侵。

(4) 核实

如图 5-4 所示。A1 报告发现攻击，J 则询问 A2 是否也检测到同一种攻击。若 A2 报告检测到同一种攻击，那么 J 就认为 A1 的入侵报告得到验证。这种情况在入侵追踪时是非常有用的。

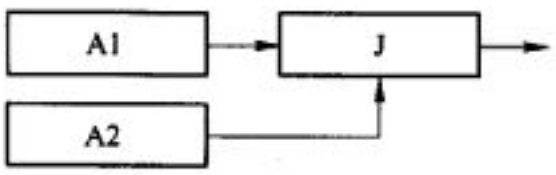

图 5-3　CIDF 协同方式：互纠

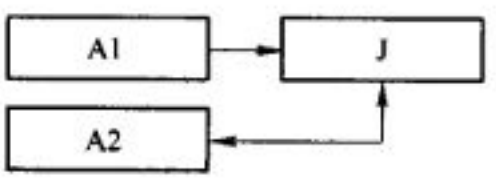

图 5-4　CIDF 协同方式：核实

(5) 调整

如图 5-5 所示。A 根据所接收到的警告信息调整监测。A 给 E 发送一个请求，询问应该监测的对象是什么。然后，A 接收 E 的回答信息，加以分析并进行监测操作。

(6) 响应

如图 5-6 所示。如果分析器判断出一个特定的处理过程正在攻击某个主机,或者是一个特定的网络连接被用来发起攻击。那么分析器应当作什么? 当然可以向管理员发出报警,但是人的响应要比机器慢。因此,入侵检测器需预先设置自动应急的脚本,以便在紧急情况下 IDS 可以直接响应。

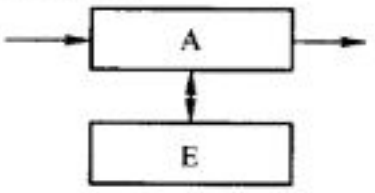

图 5-5 CIDF 协同方式:调整

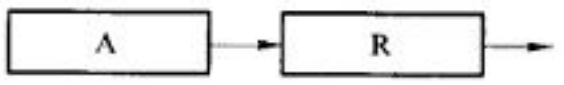

图 5-6 CIDF 协同方式:响应

2. CIDF 的公共入侵规范语言(CISL)

可以说,CIDF 最主要的工作就是不同组件间所使用语言的标准化,CIDF 的体系结构只是通信的背景。而两个组件之间的通信就像人与人之间的交流,不论所说的话的传输方式是通过空气、电话还是加密的电话,最重要的也是最有意义的就是双方能够互相理解,这也是 CISL 的意义所在。

在 CIDF 模型里,构件接收的输入流被用来驱动分析引警进行处理,并将结果传递到其他部件。换句话说,一个构件的输入流可能就是其他构件的输出。构件之间的这种输入输出关系是不同于其他系统的。尽管用来交换信息的类型是各种各样的,但事实上可以导出一种公共的数据表示方法,使所有互相连接的构件都从中获益。这意味着允许入侵检测构件引发其他的活动,如状态监视。另外可附加的功能就是应急响应。例如搜集特定的事件使得入侵检测系统内部形成反馈机制。CIDF 用通用入侵说明语言 CISL 对事件、分析结果、响应指示等过程进行表示说明,以达到 IDS 之间的语法互操作。CISL 语言使用符号表达式(简称 S-表达式),类似于 LISP 语言。S-表达式是一种抽象的数据结构,可以由原子递归定义如下:

1) 原子是 S-表达式。

2) 如果 a1、a2 是 S-表达式,则(a1、a2)也是 S-表达式。

3) 有限次使用(1)、(2)所得的表达式都是 S-表达式,此外没有别的 S-表达式。

CISL 设计的目标是:

- 表达能力。CISL 语言应当具有足够的词汇和复杂的语法来实现广泛的表达,主要针对事件因果关系、事件的对象角色、对象的属性、对象之间的关系、响应命令或脚本等几个方面。
- 表示的惟一性。要求发送者和接收者对协商好的目标信息能够相互理解。
- 精确性。两个接收者读取相同的消息不能得到相反的结论。
- 层次化。语言当中有一种机制能够用普通的概念定义详细而又精确的概念。
- 自定义。消息能够自我解析说明。
- 效率。任何接收者对语言格式的理解开销不能成倍增加。
- 扩展性。语言里有一种机制能够让发送者用使用的词汇来表明接收者的事实。或者是接收者能够利用消息的其余部分解析新的词汇的含义。
- 简章。不需理解整个语言就能接收和发送信息。

- 可移植性。语言的编码不依赖于网络的细节或特定主机的消息。
- 容易实现。

CISL 为了能够实现自定义的功能，规定每个 S-表达式都有标记，称做语义标识符(SID)，用于说明后续的 S-表达式的含义。CISL 的语义标识符有动作 SID、 角色 SID、附带 SID、属性 SID、原子 SID、连接 SID、指示 SID 和 SID 扩展名等多种类型。

3. CIDF 的通信架构

CIDF 要实现协同工作，还要解决构件之间通信方面的两个问题：

- CIDF 的一个构件怎样才能安全地联系到其他构件，其中包括构件的定位和构件的认证。
- CIDF 如何保证构件之间能够安全有效地通信。

为了解决以上两个问题，CIDF 采用了一种称为 Matchmaker 的通信架构。它提供了一个标准的、统一的方法，使得 CIDF 的构件之间互相识别和定位，让它们能够共享信息。这样极大地提高了构件之间的互操作能力，从而使入侵检测和应急系统的开发变得容易。Matchmaker 支持目录提供查询服务。中间件有两种运行方式，一种是 Matchmaker 客户程序自带中间件，如图 5-7 所示。

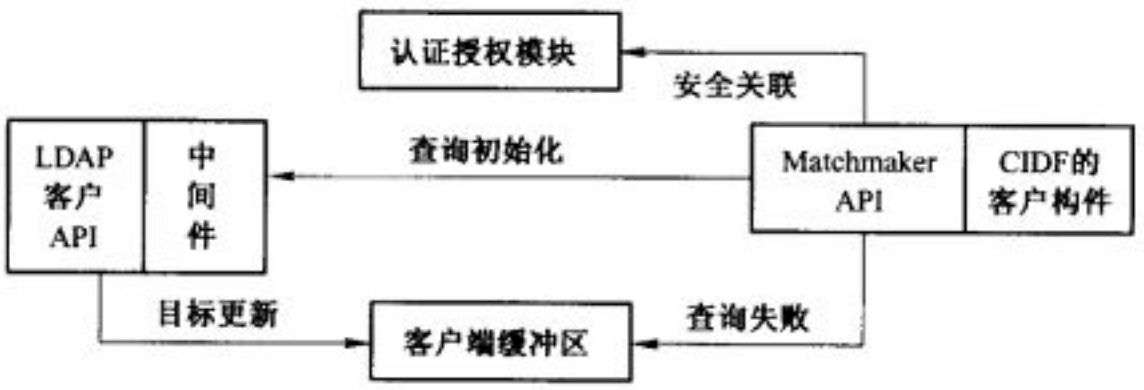

图 5-7　Matchmaker 客户自带中间件

Matchmaker 允许中间件独立于客户而存在，可以是同一主机的单独的进程，也可以完全是其他主机上的进程，如图 5-8 所示。在这种情况下，一个中间件可以为多个客户服务。

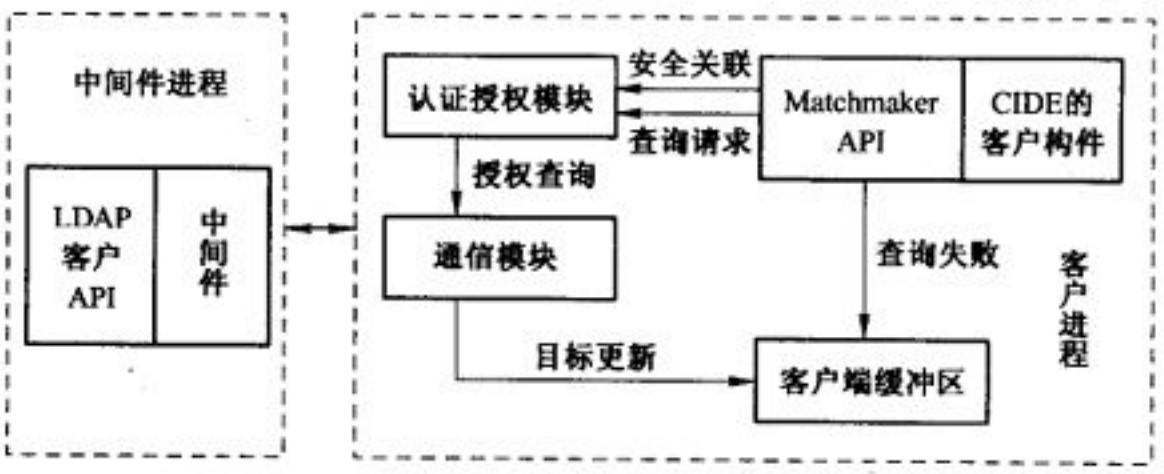

图 5-8　中间件与 Matchmaker 客户分离

4. CIDF 总结

CIDF 从组件通信入手，完成了一系列的标准化，主要包括：

- 通过组件标识查找，或更高层次上地通过特性查找通信双方的代理设施和查找协议。
- 正确鉴别、安全加密、有效的组件间通信协议。

- 定义了一种能使组件间互相理解的语言 CISL。
- 说明了进行通信所用的主要的 API。

如果完全按照 CIDF 标准化进行开发，应该说是可以达到异构组件间的通信和管理的，但是，这种标准化也有一定的不足。

- 复杂性：首先，代理设施的建立和遵循查找协议查找到对方都是非常复杂的；其次，对 CISL 语义的理解也是相当复杂的。
- 时效性：由于协议的复杂性，必然导致时间消耗过大、延迟增长。
- 协议的完整性：文档很多地方还不太完整，需要进一步细化。

目前，CIDF 小组准备加入 Internet 工程任务组（IETF），有可能使得 CIDF 成为入侵检测的标准。上述 CIDF 的内容仅仅是 Internet 草案。不过 CIDF 的重要贡献在于将软件构件理论应用到入侵检测系统中，定义了构件之间接口方法。从而使得不同的构件能够互相通信和协作。事实上，今天已有不少的 IDS 研究原型和产品，但是未考虑重用和环境变化。

5.3.2 IDWG 的标准化

为了适应网络安全发展的需要，Internet 网络工程部 IETF（Internet Engineering Task）的入侵检测工作组（Internet Detection Working Group，简称 IDWG）负责制定入侵检测响应系统之间共享信息的数据格式和交换信息的方式，以及如何满足系统管理的需要。IDWG 的主要工作如下：

- 制定入侵检测消息交换需求文档。该文档内容有入侵检测系统之间通信的要求说明，同时还有入侵检测系统和管理系统之间通信的要求说明。
- 制定公共入侵语言规范。
- 制定一种入侵检测消息交换的体系结构，使得最适合于用目前已存在协议实现入侵检测系统之间的通信。

目前，IDWG 已完成入侵检测消息交换需求、入侵检测交换数据模型、入侵告警协议基于 XML 的入侵检测消息数据模型等文档。下面选择部分文档作一简要介绍。

1. 入侵检测消息交换需求

网络安全的发展使得入侵检测系统之间迫切需要交换信息，其优点如下：

- 目前有许多商业和免费的入侵检测系统，新系统也在不断出现。其中的一些旨在检测网络入侵，一些则检测主机操作系统，还有一些检测应用程序。即便是同一类产品，功能方面也有很大差别。因此，用户很可能同时选用多个产品，需要查询来自一个或多个管理器的输出信息。事件报告的标准格式能极大地简化此项工作。
- 入侵行为经常涉及多个受害组织或同一组织的多个站点，而那些站点通常使用不同的入侵检测系统。通过关联多个站点或管理域对检测分布式入侵大有裨益。各站点的报告采取统一的格式有助于任务的完成。
- 通用数据格式的存在使不同系统的构件更易集成。入侵检测的研究成果将会更好地移植到商业产品中。
- 除了入侵检测分析器到入侵检测管理器的通信外，IDEF 通信系统也可能引发各种 IDS 构件之间的通信。上述原因表明，可疑事件报告的统一格式有助于 IDS 市场更加成功

地发展和创新，并使 IDS 的用户从安装的入侵检测系统中取得更好的效果。

然而，入侵检测系统之间必须协商一种标准的消息数据格式和通信方式，这样才能实现信息共享。入侵检测消息交换需求文档从消息格式、消息内容、通信机制、安全等方面作了较好的要求说明。基本要求有以下两点：

- 尽可能参考和使用已公布的 RFC 文档。
- IDEF 必须能够适应 IPv4 和 IPv6 的工作环境。

下面介绍 IDEF 其他方面的需求。

(1) 消息格式需求

由于网络安全和入侵检测是跨越地理、政治和文化边界的领域，IDEF 消息格式应充分支持国际化和本地化。这样 IDEF 消息格式符合表示习惯有利于操作员理解。由于一些管理器的要求，IDEF 消息格式必须允许管理器对消息数据进行过滤和聚合。

(2) 通信机制需求

IDS 管理器常常依靠接收的 IDS 分析器的数据来有效地工作。这样的 IDS 管理器可能要依赖 IDEF 消息，所以 IDEF 消息的可靠传递就十分重要。因此，IDEF 必须支持可靠的消息传递。使如 IDEF 系统可以依靠 TCP 的可靠性机制或自行设计 UDP 上的可靠性协议。

由于防火墙可能安装在 IDEF 分析器和相应的管理器之间，IDEF 消息传输要通过防火墙，为此要求 IDEF 必须支持入侵检测构件之间的消息传递穿过防火墙边界但并不损害安全性。IDEF 消息通信的建立不能削弱受保护网络的安全性。我们也不能把 IDEF 消息和其他类型的通信混合起来(如通过 HTTP POST 方法)，因为这样做会使一个组织难以将 IDEF 通信和其他类型的通信区分开来。

(3) 安全需求

由于管理器将报警消息用于指导应急响应或分析企业网络的安全，所以收发双方确信对方的身份就是很重要的。IDEF 必须支持分析和管理器之间的相互认证。

IDEF 消息可能包含入侵者十分感兴趣的极度敏感信息(如口令字)。由于它们可能经过未受控制的网段传输，消息内容的防护非常重要。IDEF 必须支持交换过程中消息内容的保密性。保密设计方案必须能支持多种加密算法，并适应各种环境的变化。

IDEF 必须保证消息内容的完整性。这是因为管理器用 IDEF 消息指导企业网络的安全，管理器确信传输后的消息内容未被修改是致关重要的。完整性设计方案必须能够支持多种完整性机制，并且必须能适应广泛变化的环境。例如 MD5 算法可能成为 IDEF 设计中的一部分。

IDEF 通信机制应能够确保原发 IDEF 消息的非否认性。假设安全敏感信息正在被交换，系统操作人员将消息和 IDEF 实体的原发方相联系是很重要的。例如，如果事后不能证实 IDEF 报警的源方实际就是源发方身份，IDEF 报警的证据价值就要大打折扣。

IDEF 通信机制应抵制对协议的拒绝服务攻击。攻击安全通信系统的普通方法是耗尽资源。虽然这不能破坏消息的有效性，但它能阻止所有的通信。IDEF 通信机制抵制这种拒绝服务攻击是很需要的。例如，攻击者渗透一个有 IDS 防御的网络。尽管攻击者不能肯定是否有 IDS 存在，但他能确信应用级的加密通信流(如 IDEF 流量)正在被攻击网络的构件之间交换。于是他隐藏并发起多个 flood 事件来破坏加密通信。如果 IDEF 能抵制这种攻击，攻击者放弃这一行为的机会就会增加。

IDEF 通信机制能抵制恶意的消息复制。削弱通信机制安全性能的通常做法是复制发送信息，即使攻击者可能并不理解它们，只是企图混淆接收者。例如，攻击者渗透一个 IDS 防御的网络。攻击者怀疑 IDS 的存在，并识别出系统构件加密的通信流，这是一个潜在威胁。虽然不能读取到这些信息，但是可以拷贝他们，并将多个副本定向到接收者，以期造成某种混乱。如果 IDEF 能防止消息复制，攻击者放弃这种攻击的几率就会增加。

(4) 消息内容需求

IDEF 消息必须覆盖以前存在和将来可能出现的各种类型的入侵检测机制，至少有下面几种：

- 基于签名的检测系统。
- 基于异常的检测系统。
- 基于相关性的检测系统。
- 基于网络的检测系统。
- 基于主机的检测系统。
- 基于应用程序的检测系统。

许多种不同类型的入侵检测系统依靠分析各种各样的数据源来进行入侵检测。一些是基于轮廓文件、操作日志、攻击签名做分析检测，一些是通过建立行为模型，另一些是通过建立行为模型，根据观测的结果与标准偏离程度大小检测异常行为，从而发现入侵者。不同入侵系统的检测报告结果数据不相同，该标准应当支持所有的数据类型。

若事件为已知的，IDEF 消息内容应指明事件的标识名。这个标识名字必须来自标准的事件列表，如果事件标识尚未标准化，则可以是某种特定实现的名字。本文表述了入侵检测消息标准化方面的需求，使得入侵检测管理器能从多种实现的分析器接收器接收警报，以及管理器对事件的攻击方法和可能补救措施的信息。有些事件很著名，这种识别对操作员很有帮助。例如，攻击者发动的攻击被两种不同实现方法的两个不同的分析器检测到。它们都向入侵检测管理器报告同样的事件标识，即使每个分析器使用攻击检测算法可能并不相同。

IDEF 消息必须包含与事件有关的任何安全建议索引，这些建议是由众多的事件响应小组、厂商和研究组织提供的，这些信息被管理员用来报告和修复出现的问题。例如，攻击者实施了一种常见的攻击，则 IDEF 消息中包含了与此攻击相关的 CERT 建议，操作员就可以利用这些信息来修补系统的漏洞。

IDEF 消息必须包含与特定事件相关的详细信息的索引。若操作员想得到有关某一事件的更多信息，就可以通过这个索引而得到特定的事件的详细细节。例如，攻击者攻击主机并被入侵检测系统检测到。IDEF 消息包含了一个指针，指向系统审计数据的若干记录项。

IDEF 消息必须包含能识别的事件源的身份和目标构件的标识。对于一个基于网络的事件，事件源和目标标识就是会话连接的源和目的 IP 地址。其他类型的事件，事件源和目标标识会有所不变化，例如那些在操作系统层或应用层检测到的攻击事件。IDEF 的这些信息将使得操作员能够识别出事件的源和目标。例如，入侵者利用缓冲区溢出发起对 DNS 服务器的攻击。IDEF 报警消息指示了作为目标的 DNS 服务器，以及发动攻击的源 IP 地址。

IDEF 消息必须支持不同类型的设备地址的表示。与入侵关联的设备可能有网络不同协议层的地址(如第 2 层和第 3 层地址)，另外，入侵相关的设备可能使用非 IP 为中心的地址。

例如，IDS 通过 IDEF 消息中设备的 IP 地址和 MAC 地址识别出特定设备上的入侵。另一种情况下，IDS 利用 IDEF 消息中的 MAC 地址信息识别出只有 MAC 地址的设备上的入侵。还有一种情况就是用 ATM 交换机的 E.164 地址格式识别入侵。

IDEF 消息必须包括指明该事件对目标系统可能造成的影响信息。事件给目标系统所造成影响的信息揭示了入侵者正试图做什么，也是操作员实施危害评估的关键数据。然而，所有系统并非都有确定的这些信息，但是可以备份数据的系统是重要的。例如，入侵检测分析器检测出缓冲区溢出攻击，若 IDEF 消息包含这种攻击信息，则表明此缓冲区溢出攻击对目标系统的影响是“试图获得 root 或管理员特权”。入侵检测操作员可以根据这条信息来提高响应的优先级。

IDEF 消息必须提供有关分析器对事件自动采取的应急响应的信息。对于管理员来说，知道是否有自动响应以及怎样响应是非常重要的。这将有助于决定是否采取进一步的行动。例如，攻击者发动攻击，入侵检测系统检测到了攻击，管理员禁止了进行可疑活动的用户的账号。可以将该用户挂起 10 分钟，以便使管理员有时间调查可疑活动。IDEF 消息中就包含了这样的消息。

IDEF 消息必须包含识别和定位出报告此事件的分析器信息，检测分析器的标识常常在决定怎样响应一个特定事件时是十分有用的。例如，一个有趣的实例是入侵事件过程涉及整个网络。如果多个分析器检测到这一同样的事件并报告，分析器的标识可能就提供了目标系统所在网络中位置的信息，并采取一种特定的响应。用 IP 地址可以识别分析器。

IDEF 消息必须包含检测出该事件的工具的名称信息。用户可能运行多个入侵检测系统保护他们的企业。这些数据将帮助系统管理员认清究竟是哪个检测工具发现了该事件。例如，实现者 Y 的 X 工具检测到一个潜在的攻击。操作员就能根据已知道工具　的能力和弱点来决定进一步的操作。

IDEF 消息必须能声明报告的可信度信息。分析器对 IDEF 的这个内容是可选的，因为并非所有的分析器都可获得这些数据，许多入侵检测系统都没有一个门限值，以决定是否报警。这就影响报告的可信度，并导致误报警。例如，警报门限值设置较低，以便能发现所有的可疑活动，而不管发现真实攻击的具体可能性。这一可信度量用于表明发生的是一个低概率还是高概率的攻击事件。

IDEF 消息必须是惟一的，以使它与其他的 IDEF 消息区分开。IDEF 消息有可能是由地理分散的多个分析器在不同时间发送。IDEF 消息的唯一标识将有利于数据推理和相关分析。例如，惟一标识符可由一个源标识（如 IPv4 或 IPv6 地址）和源发方产生的惟一序列号连接而成。在一个典型的 IDS 配置里，低层的事件分析器将分析结果报告给高层的分析器，同时将原始的检测信息记录到数据库中。在这种情况下，惟一的原始消息标识符从数据库中检索原始消息。

IDEF 必须支持每个报警事件中有产生日期和和时间。除了报警产生的日期和时间，IDEF 还可能报告检测事件时的时间。时间对于报告和相关性考虑都很重要。事件检测到的时间可能不同于报警产生时间，因为实际产生报警消息要花费一些时间。如果能决定事件发生的时间，建议将此信息置于报警消息中。例如，如果一个事件在夜深人静之时报告，操作员可能为其赋一个比发生在白天繁忙时段的相同事件更高的优先级。

报告的时间以消息产生本地时间和时区差作为根据。为了便于合件进行相关分析，管理员将统一 IDEF 报警中报告的时间格式。例如，一个分布式的入侵检测系统，有多个分析器并位于同时区，它们都将结果报告给管理中心。若一个跨越多个时区的攻击被分析器检测到，管理中心需要足够的信息统一这些警报格式、以判段这些分析器所报告的攻击是同一种攻击。

日期的报告格式必须有足够的能力将报告日期延续到2038年以后。IDEF 要有一个长的生命期，要能适合多种应用环境。因此，必须避免限制 IDEF 生命期的各种因素(如 2038 年后的日期表示限制)。例如，若 IDEF 使用 UNIX 时间作为时间表示，它将在 2038 年出问题。这种表示日期方法是不满足要求的。

IDEF 消息必须是可扩展的，允许实现者定义特定相关的数据，实现者可选择使用某种机制表示。这些数据由每个实现者指定说明，但实现者必须指明怎样去解释这些扩展。

IDEF 消息的语义定义是良好的，这样有利于理解消息表达的意义和减少错误。管理员将依据这些消息，决定采取何种操作，但正确理解消息是前提。例如，若管理员接收 IDEF 消息，却按相反的方法解释，而构造此消息的实现者则有着与操作员全然不同的另一种含义。这样，正确的动作将变得不正确了。

IDEF 报警的标准列表必须是可扩充的。随着新的事件的产生以及新的检测方法的出现，IDEF 必须能够适应技术发展。新的入侵方法产生很快，一些是现存入侵模式的变种，一些是全新的入侵技术。如果 IDEF 不具有可扩充性，那么本标准的实用性就会很快消失。

IDEF 自身必须可扩展。随着新的入侵检测技术和事件的新信息出现，IDEF 消息必须能够包括这些新消息。随着入侵检测技术的不断发展，关于检测事件的新的信息可能出现，IDEF 消息必须能够通过特定的实现方式以得到扩展，这样就能包含这些新的信息。例如，一个新型的入侵检测分析器，它能够识别攻击者的真实登录名以及记录攻击者的跳转站点的列表，那么实现者应在 IDEF 消息中包含这些新信息。

报警标识的标准列表必须是可扩展的，允许实现者和管理者根据需要而变化。IDEF 规定每种入侵的基本信息。为区分各自的入侵检测系统，不同的实现者希望提供 IDEF 以外信息的能力。此外，特殊的实现将 IDEF 用于非标准事件。例如，一个 IDS 要检测一种新事件，IDEF 标准报警标识没有包含这种事件。实现者就要通过一个私有的、专门实现的标识符发送相应的 IDEF 消息。

新的报警标识的定义和标准化过程必须独立于实现。新报警标识的定义过程不能只青睐于某种 IDS 的实现，否则，会造成负面影响。例如，实现者 A 发现了一种新的入侵事件并向 IDEF 事件过程提交有关信息。实现者 A 必须意识到这是一种积极的行为。

2. 入侵检测消息数据模型

至目前为止，已制定出来有关入侵检测的消息数据模型有两类，即面向对象数据模型和基于 XML 的数据模型。下面就分别概述一下这两种数据模型。

(1) 面向对象数据模型

将一个面向对象的模型用于 IDWG 的数据表示是由于以下原因：

- 报警的信息固有的不同类型。某些警报只有少量的信息，例发事件的来源、目标、名称以及时间等。面其他的报警信息提供更多上下文，例如端口号、服务类型、进程、用户资料等信息。因此，提出一种数据表示格式足够灵活以适应不同的需要。

- 入侵检测工具环境都不是相同的。一些攻击通过分析网络的流量进行检测，而其他的就是使用操作系统的日志、应用审计信息进行检测。相同的攻击可以用不同检测工具报告，而所报告的信息是不一样的。
- 入侵检测工具的能力不相同。安装简单的工具只提供少量的信息，而复杂的工具将会影响运行的系统，这是由于入侵检测系统提供详细的信息。为了进一步处理报警的信息，数据模型应当方便通过工具转换格式。
- 入侵检测操作系统环境是不一样的。根据使用的不同网络和操作系统，检测到的攻击具有不同的特性。数据模型应当容纳这些不同点。
- 商业厂商的目标是不同的。受操作系统环境、开发工具等限制，开发商希望传递少量关于某些攻击的信息，这样有利于产品的销售。

下面简单地讨论用面向对象方法构建入侵检测数据模型。模型由报警类、分析器类、名字类、目标类、来源类、结点类、用户类和进程类组成，这些类的关系如图 5-9 所示。

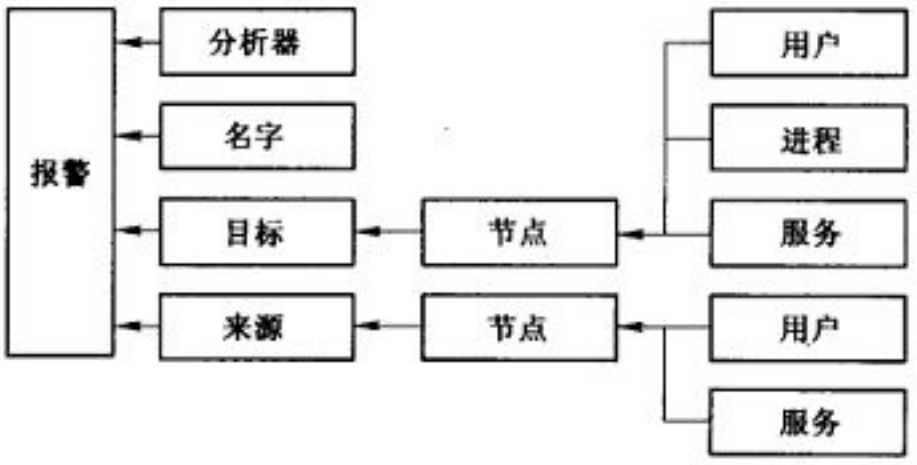

图 5-9　数据模型总体架构

报警类是模型的核心部分，每个报警都与分析器相联系，或者与一组目标、来源关联。报警类的属性包括：

- 版本号(version)，用于描述类的层次。
- 报警序列号(alertID)，要求该序列号惟一，由分析器产生。
- 可信级(confidence)，用来说明对攻击产生的警报的可信度。
- 影响力(impact)，评估警报对系统的影响。
- 方法(method)，检测器采用的方法。
- 时间(time)，警报产生的时间。
- 签名(signature)，识别攻击的特征。
- 应急(reaction)，对抗攻击的方法。

(2) 基于 XML 的数据模型

XML 是 eXtensible Markup Language 的缩写，其中文为可扩展标记语言。XML 着重于对文件信息的结构性描述。XML 由 XML 工作组制定，1998 年 W3C 正式通过推荐 XML1.0 版本。XML 本身是一种源语言，可以用 XML 创造出新的标记语言。例如 CML 是最早的一种 XML 的应用产品，主要用来描述化学分子的结构，使用 CML 来构建化学分子文件能使得计算机快速地检索到相关的对象。每一种由 XML 制定出的新标记语言或 XML 本身，都是用来

描述文件的结构信息，使得该文件能够轻易的被分析读取。由于 XML 文件具备这些优点，才被用来描述 IDMEF(Intrusion Detection Message Exchange Format)，其中文为入侵检测消息交换格式。IDMEF 明显的用途是可作为入侵检测分析器、探测器和管理员之间的数据通道，可用来传递报警信息。不过，IDMEF 在下面几个方面有用：

- 将不同的入侵检测产品的检测结果以独立于数据库的形式保存起来，然后对其进行处理，使得对入侵数据分析和活动的报告是全面的，而不是单独的某一方面。
- 能够接收来自不同的入侵检测产品的报警事件，具有实施更复杂交叉分析和相互验证的计算条件，而不受单一产品的限制。
- 单个的图形化界面能够显示来自不同的入侵检测产品的报警，使得用户从一个显示屏幕就能够监测许多产品的运行状况，因此用户只需学会一种界面窗口。
- 通用数据交换格式将使得不同的组织(用户，厂商，应急小组，法律机构)之间不仅数据交换容易，而且相互沟通顺畅。

采用 XML 的优点有以下几点：

- 使用 XML 可以方便地为入侵检测报警描述开发特定的自定义语言。也可以通过扩展这个语言来定义一个标准。
- 处理 XML 文档资料的软件工具可以很容易地得到，包括商用混合开放源代码的形式。用于过滤和验证 XML 各种工具，另外 API 的开发语言丰富，包括 Java、C、C++、Tcl、Perl、Python 和 GNU Emacs LISP。产品开发商可以很容易得到这些工具来使用 IDMEF。
- XML 满足 IDMEF，全面支持消息格式国际化、本地化的需求。XML 标准声称支持 ISO 10646 的 UTP-8 和 UTF-16 编码，从而使得 IDMEF 兼容一个字节或两个字节的字符集。XML 也支持详细说明每个元素的主要内容，因而用这些元素构造的语言能够使 IDMEF 容易适应支持自然语言处理的产品版本。
- XML 满足 IDMEF 消息格式必须支持过滤和聚类的需要。XML 和 XSL 的结合使得消息能够组合、丢弃和重组。
- 正在进行的 W3C 和其他的 XML 开发项目组将会提供支持面向对象的扩展和数据库。
- XML 是免费的，不需许可证和版税。

XML 文档类型定义(简称 DTD)说明了一个 XML 文档的准确语法。它定义了文档中使用的各种各样的标记(tags)以及标记间的相互关系，哪些标记是强制的，哪些是可选的等等。IDMEF 文档按照 XML 标准定义成良好的格式，符合正确语法的要求。IDMEF 文档必须是良好定义并且有效的，这样做的好处是统一格式和可重复使用。

3. IDMEF 总结

IDMEF 针对分析器和管理者之间的警告传输进行了标准化。下面是对其工作的分析和总结：

- IDMEF 最大的特点就是充分利用了已有的较成熟的标准，例如，使用 XML 解决数据表示问题，使用 TLS 解决数据的安全传输问题等，IAP 也是部分借鉴 HTTP 1.1 开发的，另外还使用 RFC2510 进行公钥管理。这样做的好处是显而易见的，因为已经标准化的协议是比较成熟和完备的，成熟的协议通常都有可供借鉴的代码工具，避免了自己开发协议可能要走的弯路，而且所使用协议的升级和完善自动保证了本协议的升级。

- 与 CIDF 相比较，在数据表示方面不仅在语法方面而且在语义方面都进行了详细的规定，方便了传输数据的解释，提高了解释的效率，但同时也降低了通用性，只能表达警告信息。
- 在通信方面，CIDF 提出的查找代理设施有很好的灵活性和可扩展性，IDMEF 各组件必须有通信对方的地址信息，这样效率很高。IDMEF 使用类似 HTTP 请求/应答方式进行通信，通信内容为字符流，通信双方需要进行很多字符串操作，如匹配、查找、转换等，因此相对于 CIDF 来说效率比较低。
- IDMEF 通信可能考虑到安全边界问题，支持使用代理，但正如其文档中所说的，部署代理需要谨慎。此外，符合 IAP 协议的代理也需要另行开发，这样就增加了开发任务。

5.3.3　标准化工作总结

以上入侵检测协议只是草案，至于这些协议将来是否能成为 Internet 标准现在还难以确定。按 IETF 规定，Internet 草案最长有效期为六个月，随时可能被其他文档更新、替换。

总的来说，入侵检测的标准化工作进展非常缓慢，现在各个 IDS 厂商几乎都不支持当前的标准，造成各 IDS 之间几乎不可能进行互操作。但标准化终究是 IT 行业充分发展的一个必然趋势，而且标准化提供了一套比较完备、安全的解决方案。

5.4　练习题

一、选择题

1. CIDF 是(　　)的简称。

A. 入侵检测框架标准草案　　B. 入侵检测数据交换草案

C. 安全部件互动协议　　D. 入侵检测接口标准协议

2. CIDF 定义了 IDS 系统和应急系统之间交换数据的方式，CIDF 互操作类型不包括(　　)。

A. 配置互操作　　B. 语义互操作

C. 语法互操作　　D. 通信互操作

3. 在 CIDF 中，IDS 各组件间通过(　　)来进行入侵和警告等信息内容的通信。

A. API　　B. SID

C. CISL　　D. Matchmaker

4. 在 IDWG 的标准中，目前已制定出的有关入侵检测消息的数据模型有(　　)。

A. 语意数据模型　　B. 面向对象数据模型

C. 基于数据挖掘的数据模型　　D. 基于 XML 的数据模型

5. IDMEF 使用(　　)解决数据的安全传输问题。

A. XML　　B. TLS

C. IAP　　D. RFC2510

6. (　　)防火墙不能发现而入侵检测系统可以发现。

A. 拒绝服务攻击　　B. 端口扫描

C. 蠕虫病毒攻击　　D. 局域网内非法登陆

7. CIDF 将通信机制构造成一个三层模型,(　　)不属于这三层模型。

A. GIDO 层　　B. 消息层

C. 数据链路层　　D. 协商传输层

8. CIDF 主要讨论消息的封装和传递,主要包括(　　)。

A. 配对服务　　B. 路由

C. 消息层　　D. 以上全部

9. 下列选项中属于免费的入侵检测产品的是(　　)。

A. RealSecure　　B. Entrust

C. Shadow　　D. NetRanger

二、简答题

1. 简述 RealSecure 的主要特点。

2. 简述国内厂家生产的入侵检测系统的共同特点。

3. 简述 CIDF 标准化工作的主要思想。

4. 简述 CIDF 是怎样解决构件之间的通信问题的。

5. 简述 IDWG 的主要工作是什么?

第6章　入侵检测系统的实现

本章导读：

本章以一个基于 Agent 的分布式入侵检测系统的设计与实现为例，介绍入侵检测系统的实现过程。包括系统的体系结构、主机 Agent 的设计与实现、分析 Agent 的设计与实现、中心 Agent 的设计与实现以及 Agent 之间通信的设计与实现。

6.1　系统的体系结构

6.1.1　现有入侵检测系统的局限性

现有入侵检测系统多数都采用单一体系结构，即所有的工作包括数据的采集、分析都由单一主机上的单一程序来完成，而一些分布式的入侵检测系统只是在数据采集上实现了分布式，数据的分析、入侵的发现还是由单个程序完成的，这样的结构有如下缺点：

- 可扩展性较差。由于所有工作都是由单一主机执行，被监控的主机数和网络规模受到限制，入侵检测系统的实时性要求较高，数据过多会造成其过载，从而入侵检测系统因来不及处理过量的数据或丢失网络数据包而失效。
- 单点失效。当入侵检测系统自身因受到攻击或其他原因而不能正常工作时，其保护功能就会丧失，会影响到整个系统。
- 系统缺乏灵活性和可配置性。当系统需要加入新的模块和功能时，整个系统就需要修改和重新安装。

为了有效地解决上述问题，建立一个健壮、灵活且具有良好伸展性的入侵检测系统，我们引入了 Agent 技术，提出了一个基于 Agent 的分布式入侵检测系统的架构。

6.1.2　Agent 在 IDS 中的作用

Agent 是在特定环境下能自主连续运行的软件实体，它能够从一个地方移动到另一个地方，以灵活和智能的方式适应环境的变化，并具有学习功能。

在这里所说的 Agent，是指一个在主机上执行特定安全监测任务的软件代理，是一个可独立运行的实体。它可以独立地完成某些功能，并和其他 Agent 之间相互通信，和其他 Agent 一起协作。另外，Agent 也可以受来自其他实体的高级控制命令的控制，这些命令可以是启动命令、终止命令，也可以是对该 Agent 的配置的改变。它可以在运行期间重新配置，但不需要重新启动。另外，在引入到一个更复杂的环境中之前，Agent 能够独立地进行测试。几个 Agent 可以形成一组，相互交换信息、协同工作，从而比单个 Agent 能够得出更复杂的结果。

在入侵检测中，利用 Agent 来采集和分析数据有以下主要特点：

(1) 因为 Agent 是独立的运行实体，因此，不需改变其他的组件，即可向系统中增加或从

系统中移走 Agent。例如,假如我们要采集一组新数据,或想要监测一种新的攻击,不用影响已经运行的 Agent,只需启动一个适当的新 Agent 即可。同样,不需重新启动 IDS,即可终止不再需要的 Agent,或者通过适当的命令重新配置 Agent,也可以按需要对 Agent 进行升级,只要保持它们的外部接口不变或使其向后兼容即可。这样,整个系统的升级就会变得比较容易。

(2) 如果一个 Agent 由于某种原因而停止了工作。那么,可能有以下两种情况出现:

- 如果该 Agent 是完全独立的,那么只有它本身的结果会丢失。所有其他的 Agent 都会继续正常工作。
- 如果该 Agent 和其他 Agent 一起协作,受影响的也只是相关联的几个 Agent。不管怎样,损失只局限在有限的范围内,不会造成整个系统的瘫痪,这就保证了系统的连续运行。

(3) 如果将 Agent 以分级结构的形式组织起来,可以使得系统的可扩展性和伸缩性更好。

(4) 系统开销小。Agent 的编程可以很灵活,一个主机上完成不同任务的 Agent,可以用最适合各自任务的语言进行编程,而且这些语言可以不同。设计合理的 Agent 可以利用最少的系统资源。还可以根据需要动态地启动或停止 Agent,这样也可以节省系统开销。

(5) Agent 采集数据的方法很灵活。它可以从审计记录中获得数据,也可以通过运行命令来获得系统信息,或通过查看文件系统的状态(如检查文件属性或内容)来获得信息,也可以通过从网络上捕获数据包来探测所在系统,或者从其他任何适当的数据源获取信息。因此,使用 Agent 采集数据的 IDS 跨越了基于主机和基于网络的传统界限,基于主机和基于网络的两种 Agent 协同操作,可以构造一个完整的网络防御体系,比如可以更好地阻止插入和逃避攻击。将基于主机和基于网络的检测技术集成起来,也是下一代 IDS 的发展方向。

6.1.3 系统的体系结构

该系统是一个基于 Agent 的分布式层次体系结构。整个系统由分布在网络中的多个功能 Agent 组成,各个 Agent 分别位于不同的主机上,Agent 之间既可以独立工作,又可以相互协作,形成一个统一的层次体系。每台主机上可以有多个独立运行的具有不同功能的 Agent,Agent之间可以相互通信、协作。所有 Agent 按功能分为四类:中心 Agent、分析 Agent、主机 Agent 和网络 Agent。

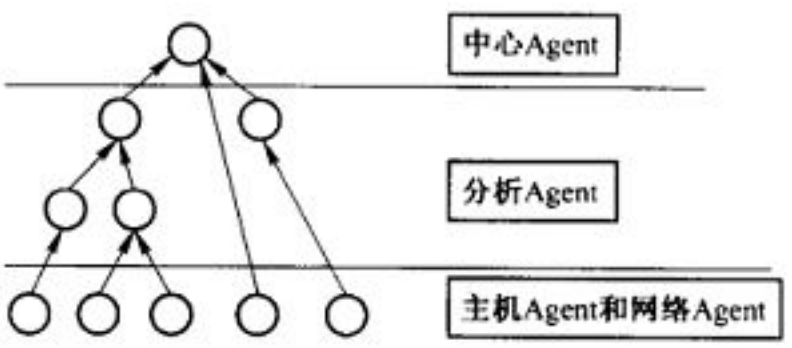

图 6-1 系统的体系结构

其体系结构如图 6-1 所示,图中各层的功能如下:

第一层:中心 Agent。它着眼于整个网络,在高层对整个系统进行监控,也是系统管理员管理整个 IDS 的工具。主要完成的功能有:对分析 Agent 不能判定的情况进行判定,拥有更高的智能和安全性;对系统中的所有 Agent 进行统一的配置和管理;显示告警信息并进行响应处理。

第二层:分析 Agent。它收集主机 Agent 和网络 Agent 上报的信息,对它所覆盖网段的整个情况进行综合评估,判断是否有入侵行为发生。分析 Agent 注重于高层次的分析方法,综合运用概率统计、神经网络、数据挖掘等技术,进行特征提取和模式匹配,不断学习,动态扩充入

侵模式库，动态更新系统正常行为轨迹，以便发现异常入侵。分析 Agent 是整个入侵检测系统的大脑，分析方法则是该系统的思维能力。各种分析方法都有各自的优势和不足，因此，系统中分析方法应该是可以动态更换，并且多种算法可以并存的。

第三层：主机 Agent 和网络 Agent。主机 Agent 在所监控的主机上以各种方法收集信息，包括系统日志、监视用户行为、系统调用、该主机的网络通信等。再对这些数据进行初步的分析和处理。网络 Agent 负责收集网络中的原始分组数据包，并对数据进行分类和过滤，并从其中寻找可能的入侵信息或其他敏感信息。它们都具有数据分析功能，对于已知的攻击，可采用模式匹配的方法来检测，这样可以大大提高系统的处理速度，也可以减少分析 Agent 的工作量及网络传输对系统的影响。

6.1.4　系统策略

为了保证该系统的安全性、完整性，制定了以下系统策略：

- 自上而下的逐层控制机制。上层可以控制下层实体。同层实体之间可以相互发送事务信息，但不能改变另一实体的机制。这样就防止了破坏的扩散。
- 安全通信机制。采用加密技术和认证机制保证 Agent 之间的安全通信。
- 可扩展性。利用 Agent 的自治性和系统的层次性保证 IDS 的规模可扩展；采用统一的框架设计入侵检测模块，其规则可以扩展，可以引入其他同标准的 IDS 协同工作。
- 健壮性。分析 Agent 的增减不会影响整个系统，主机或网络 Agent 的失效也只会影响其自身节点。
- 系统恢复能力。各个 Agent 都有系统映像检查机制保证其安全性，一旦发现某 Agent 失效，将主动向上层 Agent 发出信息，由上层的 Agent 进行恢复。

6.1.5　关键技术分析

系统涉及的关键技术如下：

1）Agent 之间的通信要考虑通信开销、可靠性和安全性问题。对入侵检测系统而言，消息的机密性和认证也是很重要的。Agent 之间的通信可以采用安全通道进行数据交换，防止数据被修改、插入或删除。由于 UDP 不能保证消息传递的可靠性，我们采用 TCP 作为 Agent 之间消息传递的方式。

2）系统的负载问题。在整个入侵检测系统中涉及到大量的数据处理和数据传输，因此在设计实现系统的过程中要充分考虑大量数据传输的高效性和实时性，平衡各个 Agent 之间的数据处理和通信。

3）分析方法的选择。不同的 Agent 在系统中实现不同的功能，所以要针对不同的 Agent 采取不同的分析和实现方法。目前误用检测技术有专家系统、模型推理、状态转换分析算法等，异常检测有统计分析、神经网络、数据挖掘、遗传和免疫等算法。目前各种方法都各有其优缺点，该系统拟将综合利用各种分析方法，提高系统的性能。

6.2 主机 Agent 的设计和实现

主机 Agent 处于整个分布式系统的最低层。它的功能是在主机上以各种方法收集信息并对这些数据进行分析，包括分析日志、监视用户行为、分析系统调用、分析该主机的网络通信等，所以主机 Agent 是依赖于操作系统平台的，在此我们选用的是 Linux 操作系统。我们根据 Linux 操作系统开放源码的特点，通过安全性分析，提出了基于 Linux 内核的主机Agent的设计和实现方案。

6.2.1 Linux 安全性分析

目前，Linux 操作系统的安全级别较低，相应的安全增强软件也不丰富。通常，提到安全问题人们很容易想到防止黑客入侵，而很少考虑如果系统已被入侵怎样使系统损失降低到最小，并使黑客无法安装后门，防止下一次入侵。系统入侵的手段是丰富多样的，而当前的Linux系统在安全方面的工作有许多不足之处，这种现状很有必要改进和完善。

1. **不安全因素**

现在，在 GNU/Linux 系统中不完善之处主要表现在以下几个方面。

- 超级用户可能滥用权限。超级用户，可以做任何事情，包括删除不该删除的系统文档、杀死系统进程以及改变权限等等。
- 系统文档可以被任意地修改。在 Linux 系统中，有许多的重要文件，比如/bin/login，如果入侵者修改该文件，就可以轻易地再次登录。
- 系统内核可以轻易插入模块。系统内核允许插入模块，使用户扩展 Linux 操作系统的功能，使 Linux 系统更加模块化，但这样做是十分危险的。模块插入内核后，就成为内核的一部分，可以做原来内核所能做的任何事情。
- 进程不受保护。

2. **开后门的 14 种办法**

在一般的 Linux 计算机中开后门的方法有以下 14 种：

- 破解计算机账号密码。
- .rhosts 文件使特定用户从特定主机登录不需要密码。
- 以具有跟源文件一样时间戳的特洛伊木马程序版本来代替二进制程序。
- 替换 login 程序，提供特殊口令隐身登录。
- 替换 Telnetd。
- 替换网络服务。
- Cronjob 定时运行后门，入侵者每天在该时刻可以访问。
- 替换共享函数库。
- 替换内核。
- 在文件系统中隐藏后门。
- 在启动区内隐藏后门。
- 隐藏进程。

- IP 数据包后门。
- 在 .forward 文件中放置命令。

事实上，目前已知的后门基本上都不超出上述分类，分析以上各类后门可知，后门可以分为永久性后门和一次性后门。永久性后门是指系统重启后还能继续起作用的后门，一次性后门是指仅在本次运行时有效、重启后就无效的后门。要设置永久性后门，一定要对重要的文件进行改动，而一次性后门很可能是对重要进程进行改动。

3. 阻截黑客入侵的良方

针对以上提出的设置黑客后门的方法，我们又参考了 LIDS 系统的思想方法，提出了一种简单的增强系统安全、防止黑客入侵计算机的方法。主要思想是根据各种可能的设置后门的方法，堵截黑客后门，在最大程度上防止黑客入侵，以及在已经被入侵后最大程度上减少恶意破坏的损失。

我们可以通过以下途径来提高系统的安全性：

- 保护重要的文件。保护某些重要的文件，使这些文件具有相应的功能。如使文件在某些情况下不能被删除，或者使某些文件不能被修改，即使是超级用户也不行。
- 保护重要的进程。保护某些进程，使之不能被删除，即使用户使用命令 kill－9 也是不行的。
- 对内核进行封装。保护内核，使用户不能对内核进行模块插入。
- 实现的关键就是限制系统管理员的权限，使其权限的使用处于保护之下，即使误操作或蓄意破坏，也不至于对系统造成致命打击。

4. 增强内核级安全

增强 Linux 内核级安全的具体做法是用加载模块的方法修改和安全有关的几个系统调用。对于文件保护，在被修改的系统调用或被调用时，先检查文件的保护类型，若没有保护或属于非保护类型，则返回原来的系统调用。反之，根据文件的保护类型和用户打开文件的模式来选择打开模式，或返回错误类型。例如，对于被列为只读保护的文件，如果用户以只读模式打开文件，则返回原来的系统调用；若对只读保护的文件试图以写的模式打开，则返回错误。对于进程保护，为了保护重要的进程，使之不能被删除，可以在进程的标记位上设置一个未被操作系统使用的标志位来保护重要的进程，并替换 Kill 调用，在真正执行 Kill 调用前先检查标志位，系统将拒绝用户删除设置保护位的进程。对内核进行加固后，应禁止插入或删除模块，从而保护系统的安全，否则入侵者将有可能再次对系统调用进行替换。我们可以通过替换 create __ module()和 delete __ module()来达到上述目的。另外，对这个内核进行加固模块时应尽早进行，以防系统调用已经被入侵者替换。

修改系统调用可以通过两种方法来实现。第 1 种是直接修改系统的核心代码，然后重新编译生成新的核心。该方法的缺点是：每做一次修改都需要对系统进行重新编译，这给新核心代码的调试带来了相当大的困难。若系统管理员需要针对不同用户进行相应的配置，重新编译的工作量是巨大的，而且编译过内核的人都知道，编译过程中有非常多的选项，要编译出一个性能优良的内核非常困难。第 2 种方法是将对系统的修改内容做成一个模块，通过静态或动态地加载和卸载，该模块会修改系统调用入口。应用模块技术，可以减小系统核心代码的规模，而且在需要时才装入模块可以减小系统所占用的硬件资源，从而提高系统的性能。模块的

代码在装入核心后与核心中其他代码的地位是相同的，代码的调试就方便得多了。若管理员针对不同用户进行相应的配置，只需修改模块配置文件或在装载它时传递的参数。

6.2.2 设计思路

主机 Agent 在我们的系统中处于最低层，在主机上以各种方法收集信息，包括系统、网络、数据及用户活动的状态和行为，并对这些数据进行初步的分析，将处理后的结果传送给分析 Agent，分析 Agent 再对各个主机 Agent 传送来的数据进行综合、深入的分析。由于主机 Agent 是依赖于操作系统平台的，我们选用的是 Linux 操作系统，根据 Linux 的特点，通过对 Linux 操作系统的安全性和日志信息等进行分析后，提出了基于 Linux 内核的主机 Agent 的设计和实现。主机 Agent 主要基于以下思路来设计：

1. Linux 作为平台

选择 Linux 平台，主要有两方面的原因：

(1) Linux 是一个开放源码的平台，有利于在研究的过程中深入技术细节，由于 Linux 及其上面的大量应用都是基于开放源码，有很多黑客在其上进行了大量的工作，可以说 Linux 上的网络攻击水平代表了整个网络攻击的最高水平。

(2) Linux 是一个类似 UNIX 的系统，同时也是在 Internet 中大量使用的操作系统平台，选择 Linux 作为研究平台是非常具有代表性的，在 Linux 平台上取得的经验可以非常容易地移植到其他 UNIX 或者类似 UNIX 的平台上。

2. 基于内核的实时检测

目前人们对入侵检测提出了各种各样的解决方法。但是目前在这一领域的研究主要集中在入侵发生后才检测到入侵，而不是在入侵发生时就能判断出来，并加以阻止。这样，即使已经发现了入侵者的入侵踪迹，他也已成功地完成了入侵，窃取了他所需要的数据，达到了他的目的。所以我们需要做的是，在发现入侵的可疑迹象时及时制止入侵者，不让入侵者得逞，实现真正的实时入侵检测。因此我们在主机 Agent 里实现了基于内核的实时检测，它对关键的系统调用进行实时检测，当系统调用发生时，根据已知条件进行判断，如果发现操作是不合法的，将记录错误信息，并直接返回错误类型。否则将转去执行相应的系统调用。从而真正实现了实时的检测。

3. 基于内核的审计信息

由于目前市场上有多种 Linux 版本，每个版本的 Linux 所产生的日志和审计信息的内容和格式都是不相同的，那样就需要不同版本的主机 Agent 分布在不同的 Linux 的主机上，来对这些审计信息进行融合，以便于将来进行分析，这样势必会增加系统的复杂性，且难于实现。另外从非常庞大的日志文件中发现入侵者的入侵的蛛丝马迹也是比较困难的。我们根据所有 Linux 版本的内核都具有这一特点，通过内核来直接记录审计信息。这样就省去了对各种不同的日志文件进行综合，除去冗余信息的过程。而且通过这种方式得到的审计信息都是我们所关心的调用信息，没有太多的冗余信息，审计信息的格式也是统一的，这样便于将来对数据的分析。

4. 对审计数据进行分析

当从内核得到记录下来的审计信息后，需要对这些数据进行分析。考虑到主机 Agent 是

分布在网络的各个主机上的，不能给主机增加太大的负担，所以在实现时我们只采用模式匹配的方法。对审计数据进行分析时我们采用了两种模式，一种是基于目标的，一种是基于内核的。在基于目标的模式下，可以根据用户自定义的规则库来对审计数据进行分析，然后把分析结果传送给上级 Agent；在基于内核的模式下，可以定义我们所关心的系统调用，则数据分析的结果是我们所关心的系统调用的审计信息。在这两种模式下，都能够得到非常可靠的、丰富的、详细的统计信息。

5. Agent 之间的通信

Agent 之间的通信主要完成两个功能：一是将主机 Agent 对审计数据的分析结果发送到上级 Agent；二是上级 Agent 对主机 Agent 的配置，包括对主机 Agent 规则库的改变，重新启动主机 Agent 等。入侵检测系统本身是为了发现和阻止黑客攻击的，而 Agent 之间传递的数据都是非常重要的，所以一定要保证 Agent 之间数据通信的安全性。另外由于 Agent 之间需要传递审计信息，可能会需要大量的数据，这势必会增加网络的负载，所以我们在 Agent 之间传递数据时，可以先对数据进行分析，把相对重要的数据传递到上级 Agent，这样会减少网络的负载。

6. 可动态加载

Linux 中的可加载模块是 Linux 内核支持的动态可加载模块，它们是核心的一部分，但是并没有编译到核心里面去。模块可以单独编译成为目标代码，它可以根据需要在系统启动后动态地加载到系统核心之中。超级用户可以通过 insmod 和 rmmod 命令将模块载入核心，或从核心中将模块卸载。若在调试新核心代码时，采用模块技术，用户不必每次修改后都需重新编译核心和启动系统。

基于内核的实时检测和基于内核的审计部分是通过可动态加载的模块来实现的，该模块修改了系统调用入口，对关键的系统调用进行检测，及时阻止入侵者的不合法的行为，并记录详细的审计信息。应用可动态加载的模块技术，可以减小系统核心代码的规模，而且在需要时才装入模块可以减小系统所占用的硬件资源，从而提高系统的性能。使用信号机制实现进程之间的通信。这样设计的目的一方面在于通过动态可加载模块和守护进程可以占用比较少的资源完成所要达到的功能，有利于提高效率；另一方面这样的设计对于 Agent 以及整个系统来说比较稳定，一般情况下不会由于内在的因素造成系统本身的崩溃，有利于提高系统的安全性和稳定性。

6.2.3　主机 Agent 的结构

主机 Agent 的结构如图 6-2 所示。

1. 实时检测

实时检测是对 Linux 内核的关键的系统调用进行检测，判断用户的行为是否合法，如果不合法，返回错误类型，如果合法，则直接去执行相应的系统调用，它是通过可动态加载模块来实现的。在这一部分我们给出了专门针对防范缓冲区溢出攻击的解决方法。缓冲区溢出攻击是目前黑客最常用的攻击手段。其目的在于扰乱具有某些特权运行程序的功能。这样可以让攻击者取得程序的控制权，如果该程序具有足够的权限，那么整个主机就被控制了。

Linux 的内核是完全用 C 语言编写的。由于 C 语言没有数组的边界检查，这就使得数组越

界的内存溢出成为可能。一些恶意的用户利用C语言的这个特点，通过数组越界的方式，破坏了内存中的数据，尤其是函数返回时，返回的是另一段程序代码的指针，如果入侵成功的话，他将得到超级用户的权限，在目标机器上为所欲为，这是非常可怕的。Linux系统组件，如命令、守护进程和C语言库函数在调用sprintf或strcpy等函数时并没有进行必要的数组越界检查，这就很可能会导致内存溢出。这个漏洞使得某些恶意的用户有了可乘之机，他们往往利用缓冲区溢出来获得超级用户的权限。所以为了避免潜在的缓冲区溢出攻击，我们必须阻止任何超级用户进程执行危险的代码，因此我们对一些关键的系统调用execve，setuid，chown进行实时的检测，及时阻止了缓冲区溢出攻击的发生。

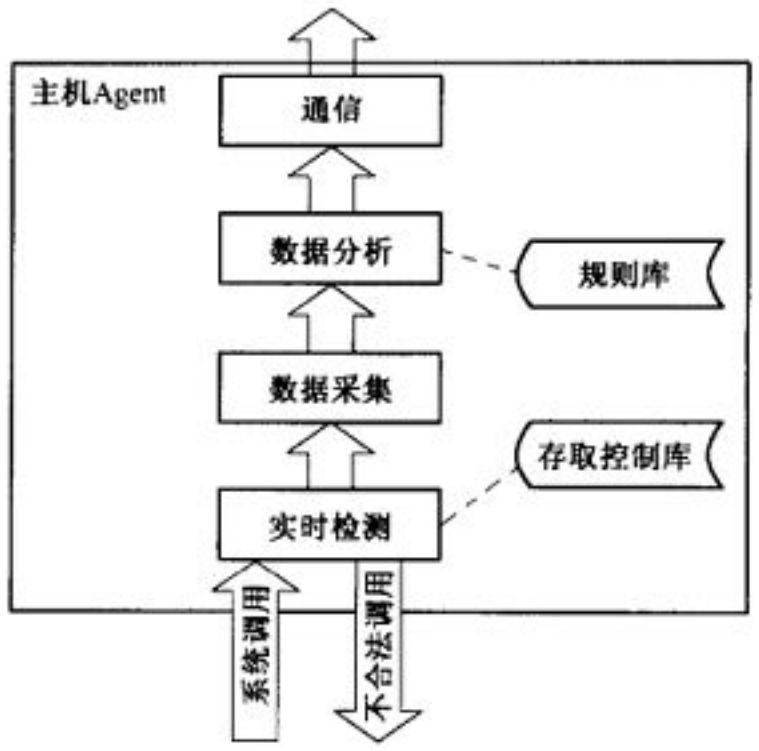

图6-2 主机Agent的结构

2. 数据采集

数据采集是对Linux内核的系统调用进行事件审计，然后将审计数据传送给数据分析模块。它封装了一些关键的系统调用，如execve(执行一个命令)，open(打开一个文件)，mkdir(创建一个目录)等。它实时地收集用户执行的进程、试图执行的进程或是有问题的系统调用的信息，然后把这些信息存放在一个临时的缓冲区中供数据分析模块使用。因为审计事件可由任何或所有激活的处理器产生，所以内核审计模块必须缓存这些审计事件直到审计守护程序获得处理器的时间片从缓存里提取数据。为了更好地满足审计传输的有效性的目标，更好地使用系统的资源，我们使用的方法是将事件的详细信息存储在一个动态的数据链表里。当系统资源变得紧张的时候，内核将放慢其他任务的执行，数据分析子模块将从缓存里读取数据，然后释放内存空间。

数据采集可以通过可动态加载的模块来实现，当主机启动的时候，它也会自动加载，这样就可以实现主机启动时就可以对主机进行实时监控。数据采集模块是对Linux系统中最底层的原始系统调用进行审计，通过对这些系统调用的分析，对它们所操作对象的操作的合法性、正确性等相关安全方面的内容进行检查，并将相关的参数、数据及信息传递给上层模块，进行其他处理。

3. 数据分析

数据分析从数据采集的临时缓冲区中读入数据，把二进制的审计数据转换为ASCII形式，然后将这些信息按一定的格式存储。它对数据采集子模块发送来的审计数据进行分析，过滤掉那些根本不存在任何攻击的数据，将可疑的数据按照用户的定义分别划分为五类告警事件：严重警告、优先警告、警告、一般信息和无关信息。用户可以选择将哪类告警事件传送给上级Agent。

对审计数据的分析我们提供了两种模式，一种是基于内核的，另一种是基于目标的，用户只能选择其一。基于内核的审计是根据用户选择对内核的系统调用进行审计，没有任何形式

的匹配。在基于目标的模式下,用户可以定义自己的规则库,它可以根据系统调用的类型、关键字、匹配方式、执行的用户来定义各种不同级别的告警事件。

6.2.4　主机 Agent 的具体实现

1. 实时检测

(1) 实现机理

目前在实时检测中只是实现了防范缓冲区溢出攻击。针对缓冲区溢出攻击的特点,我们提出了相应的解决办法。为了防范缓冲区溢出攻击,必须要阻止任何超级用户进程执行危险的代码。一个超级用户进程可以划分为 3 类:

- 交互的:这是由超级用户启动的标准的进程。用户 ID 和有效用户 ID 都为 0。由于启动该进程的用户已经对整个系统有了完全的控制权,所以这种进程不会产生特别的危险。
- 后台的:通常情况下,在系统启动的时候或 cron 守护程序以 root 的身份定时启动一些守护进程的时候,这些守护进程在某种程度上说是存在潜在的缓冲区溢出攻击的,虽然这需要比较复杂的技巧。
- Setuid:通过它可以将进程的有效用户 ID 设为 0,即超级用户 ID,这样普通用户就可以以超级用户的身份来执行程序。这时普通用户 ID 大于 0,有效用户 ID 等于 0。所以我们可以很容易判断一个进程是不是 setuid 进程。

为了简化问题的复杂性,我们只在此讲述 setuid 进程防止缓冲区溢出攻击。一旦我们描述完如何在内核识别这类的进程之后,如何防止后台类进程的缓冲区溢出攻击就变得很明显了。

在一个特定的 UNIX 系统上,总会有超级用户的程序在后台执行。在大多数情况下,系统管理员并不直接运行它们(通过交互的方式)也不控制它们的行为。所以,就像前面所提到的,这种有特权的程序存在很大的潜在性缓冲区攻击。

下面是对它们的分类:

- 在系统初始化时启动的守护进程。网络服务如 inetd, web server, mail server(尤其是 sendmail)经常以这种方式启动。还有 syslogd 守护进程。
- 由 inetd 服务启动的一些网络服务实现远程请求,如 telnet, ftp 等。
- 由 cron 守护进程定时执行的一些程序。cron 守护进程在系统启动时启动。
- 以交互的方式运行的程序(如命令行)。这主要是为了测试的目的,或者重新启动由于某种原因而停止的守护进程。

这些程序通常来说没有控制终端。为了识别这些程序,我们采用了下面的方法:

```
#define IS__A__ROOT__DAEMON(proc)!((proc)->euid)&&((proc)->tty==null)
```

在这里,前一部分是判断这个进程是否以超级用户的身份执行,后一部分是判断这个进程是否有一个控制终端。如果该进程是属于超级用户的,而且这个进程没有控制终端,则说明该进程属于守护进程。

(2) 实现了的系统调用

为了防止缓冲区溢出攻击所带来的危害,用户进行以下系统调用时需要做额外的检查:

- execve(executable—file,...):这个系统调用允许具有潜在危机的 setuid 进程去调用一个交互的命令,对关键文件进行操作。为了阻止这种情况和其他类似的攻击产生,我们在真正的系统调用 execve 执行之前,做了一些相应的检查。首先判断调用的进程是否是超级用户权限,如果不是超级用户权限,我们不需要继续做其他的工作,继续系统调用,如果调用的进程是超级用户的权限,例如它属于 setuid 进程,我们将对它做更深一步的处理,通过存取控制库来判断整个系统调用是否合法,是否可以继续进行。我们在存取控制库里保存着所有关键文件的合法和不合法的 setuid 调用进程,可以根据合法的 setuid 调用进程来判断,如果调用进程在列表内,则它是合法的,否则为不合法。也可以根据不合法的 setuid 调用进程来判断,如果调用进程在不合法的 setuid 列表内,则它是不合法的,否则是合法的,两者只能选择其一。如果判断的结果是合法的,系统调用将继续进行。否则调用终止,返回一个错误代码,并且将调用的错误信息写入临时缓冲区里。
- Setuid:我们认为任何以超级用户身份运行的进程,不论是 setuid 到超级用户的还是后台的,对目标主机来说都是一个潜在的攻击。对于这样的进程,我们也要像 execve 系统调用那样对它进行特别的检查。前面讲过 setuid 进程的特点,恶意用户往往先执行 setuid 获得超级用户权限,再去进行其他的系统调用,这样可以逃过权限检查。为了填补这个漏洞,及时阻止可疑的攻击,我们将在用户执行该调用前添加一个特殊的检查。
- Chmod:当被一个 setuid 进程调用的时候,chmod 进程可以给任何用户对关键文件(像 password 文件)写的权限,这是非常危险的。为了防止这种攻击和其他相关的攻击,当调用 chmod 系统调用时也要附加特殊的检查,防止利用 setuid 进程来修改正常文件和目录的权限。
- Chown:当被一个 setuid 进程调用的时候,chown 可以将一个可执行文件的所有者改为超级用户。chown 和 chmod 在一起用时非常危险,可以很容易得到超级用户的权限。基于这个原因,在执行 chown 系统调用的时候也需要检查,防止 setuid 进程修改正常文件和目录的所有者。
- Chgrp:基于同样的考虑,我们也要对 chgrp 进行相应的处理。

(3) 主要的数据结构和函数

首先从存取控制库(Access Control Database)讲起。

存取控制器包含每个需要侦听的系统调用的检查的合法条件。目前我们仅提供了两种类型的数据结构,setuid __ acd 用来检查 setuid 系统调用是否合法。Execve __ acd 用来检查 execve 系统调用是否合法。关于数据结构如下:

1) setuid __ acd 仅仅包含被加密的超级用户的密码,将它保存在内核内存里。当用户调用 setuid 时,它用来作为更进一步的认证。

2) Execve __ acd 包含两个 eflst __ t 类型的数组:

- 合法的:在这个数组里存放着每个关键文件的合法的 setuid 调用进程,当一个 setuid 进程执行 execve 对该文件进行操作时,如果该进程在数组内,则它是合法的,否则是不合法的。
- 不合法的:和上一个数组相反,在这个数组里存放着每个试图调用 execve 的不合法的

setuid 进程，当一个 setuid 进程执行 execve 对该文件进行操作时，如果该进程在数组内，则它是不合法的，否则是合法的。

下面讲述相关的系统调用的附加检查：

1）execve 系统调用，Check __ rootproc()函数首先判断 execve 系统调用的 setuid 进程是否是超级用户进程，如果不是，则继续调用 execve，否则根据存取控制库来判断该 setuid 进程是否合法。如果该调用不合法，它可能会有两个返回值：

- EXENA：调用进程没有被授权执行该操作。也就是说 setuid 进程的名字没有在存取控制库的合法列表中，或该名字在不合法列表中。
- EFNA：调用进程已被授权执行该程序，但是文件不被授权，例如文件的修改时间或大小不匹配。

2）setuid 进程的验证和 execve 类似。用户运行 setuid 程序试图调用 setuid(0)将程序的有效用户 ID 设为 0，则需要输入超级用户的口令。用户输入的口令和存储于内核内存的口令进行比较，如果相同，则继续执行，否则返回错误。我们期望当用户执行命令 su 时，做进一步的检查。

Chmod 和 setuid 最大的区别是不需要输入超级用户的密码，但是必须要对文件的参数做相应的检查。

Chown 和 chgrp 系统调用与 chmod 系统调用所需要的检测类似，在此不在赘述。

2. 数据采集

(1) 实现机理

我们知道，对于 Linux 这样由 UNIX 派生出来的操作系统而言，它提供的各种功能是以命令的方式实现的，由于系统本身的开放性和脆弱性导致了一些安全漏洞，因此所有这些命令的实现都将归结到系统底层的系统调用。基于这一考虑，在设计时，选择了最常用、与系统安全最密切相关的 34 种系统调用进行检测。

首先，数据采集模块是一个内核动态可加载模块，即以动态可加载模块的形式存在于系统内核空间中。由于 Linux 系统在设计时分为内核空间和用户空间，因此，一般情况下无法得到系统内核空间的数据。这里为了得到内核空间的数据，需要在/proc 目录下注册一个文件 audit，并且要为这个文件指定自定义和编写的相应的读写操作，通过调用自定义的读文件函数，可以得到 Linux 内核的全部相关信息。当 auditd 守护进程通过先前自定义的 ioctl 函数将配置信息传给/proc/audit 文件以后，audit __ module 动态可加载模块就开始根据配置信息检测所需要的系统调用。

在 Linux 系统中，当一个系统调用发生时，它会触发一个中断。在 Linux 系统中同样存在 256 个中断向量，其中 0x80 中断向量作为系统调用的总入口地址。通过总的入口地址以及具体的系统调用，在系统定义的 sys __ call __ table 中找到对应这个调用的中断服务程序的入口地址，转而去执行对应的服务程序，也就是去实现这个系统调用所要完成的功能。在对这个系统调用进行检测时，采用了类似于“钩子”的方法，即是根据具体的系统调用在 sys __ call __ table 中修改对应的中断服务入口地址，将它指向自己定义和编写的函数，在自己的函数中，将得到这个系统调用的相关参数和信息，其中重要的一项是预先对系统调用的简单分类，例如，目前简单的分为：文件相关类(open、creat、rename 等系统调用)，文件控制类(chown、chmod 等系统调用)，可执行类(execve、setuid、setgid、exit 等系统调用)，管理类(reboot、chroot 等系统调

用),根据分类进行检测分析,将大大提高效率。

从上面的阐述已经可以清楚地看到对 Linux 主机进行检测的大致思路及实现的方法。另外,还有一些内核的机制同样也起到了十分重要的作用。一是 Spinlock 机制,又称内核锁机制。由于在检测系统调用时我们需要修改中断服务程序的入口地址指向自己的函数,这时内核的某些进程会在内核空间中定义的一些变量和结构中写入相关的信息,但同时可能也会有一些进程正在读取相关的信息,这样势必造成系统错误。为了避免这种情况的发生,在一个进程读的同时,其他进程不能再写,因此,这里采用 Spinlock 机制,确保内核之间进程的同步和数据的一致。

同样,在系统总体设计时,考虑到检测系统调用在一定程度上会影响整个系统的效率,而且对于 Linux 系统而言,检测每一个系统调用产生的审计数据的数据量很大,有较大的数据冗余,分析处理不方便。因此,为了提高效率,降低冗余,这里采用了等待队列的机制。在检测时将所要检测的用户自定义事件放入等待队列中,当队列达到一定长度,它会发送信号给数据分析模块,数据分析模块将读取数据,释放所占用的内存。

所采集的审计事件的信息有:

时间(time),事件(event),实际用户 ID(realUserId),实际组 ID(realGroupId),有效用户 ID(effeciveUserId),有效组 ID(effectiveGroupId),目标用户 ID(targetUserId),进程 ID(processId),进程名称(processName),参数(arguments),文件(file),文件属性(fileAttributes),目标文件(destinationFile),新用户(newOwner),新组(newGroupOwner),源 IP(sourceIp),源端口(sourcePort),目的 IP(destIp),目的端口(destPort),返回值(returnCode)。

(2) 主要的数据结构和函数

1) 重要数据结构说明

● static spinlock __ taudit __ spin __ lock;

Spinlock(内核锁定)数据结构。

● static struct task __ struct * auditdaemon __ task __ struct;

Linux 系统进程的数据结构。

● extern void * sys __ call __ table[];

Linux 系统调用中断向量表。

● struct proc __ dir __ entry * ProcEntry;

在/proc 目录注册文件的入口。

● static struct file __ operations file __ ops=

```
{
read:auditmodule __ read,              // read
ioctl:     auditmodule __ ioctl,       // ioctl
open:      auditmodule __ open,        // open
release:   auditmodule __ close,       // release
};
```

为/proc/audit 指定的文件操作(Linux 的内核版本为 2.4)

2) 主要函数简要说明

● int init __ module(void);

初始化内核动态可加载模块,参数为空,返回值为整型数 0。

● void cleanup __ module(void);

清除内核动态可加载模块,参数为空。注销先前在/proc 下注册的文件的读写权限,使守护进程不再读取数据,释放资源,退出。

● void audit __ on(int auditnum);

对指定的系统调用进行检测,参数为指定的系统调用。通过参数 auditnum 数值所对应的具体调用,修改中断服务程序入口地址,进行检测。

● void audit __ off(int auditnum);

对指定调用停止检测,参数为指定的系统调用。通过参数 auditnum 数值所对应的具体调用,恢复原先的中断服务程序的入口地址,完成系统调用所应完成的功能。

● static int auditmodule __ open(struct inode * node,struct file * the __ file);

为/proc/audit 指定的打开文件操作,auditd 守护进程可以通过调用此函数打开/proc/audit 文件。参数 1 是文件 i 节点,参数 2 是文件指针。

● static ssize __ tauditmodule __ read(struct file * file,char * filebuffer,size __ tlength,loff __ t * ppos)

为/proc/audit 指定的读文件操作,auditd 守护进程可以通过调用此函数读/proc/audit 文件。参数 1 是文件指针,参数 2 是读缓冲区,参数 3 是缓冲区长度,参数 4 是偏移量。

● int auditmodule __ ioctl(struct inode * node,struct file * the __ file,unsigned int command,unsigned long arg);

为/proc/audit 指定的文件控制操作,auditd 守护进程可以通过调用此函数对/proc/audit 文件进行控制操作。参数 1 是文件 i 节点,参数 2 是文件指针,参数 3 是控制命令,参数 4 是传递的命令参数。

● static int auditmodule __ close(struct inode * node,struct file * the __ file);

为/proc/audit 指定的关闭文件操作,auditd 守护进程可以通过调用此函数关闭/proc/audit 文件。参数 1 是文件 i 节点,参数 2 是文件指针。

● asmlinkage int audit __ open(const char * pathname,int flag,mode __ t mode);

修改入口地址后为检测系统调用而定义的新的“钩子”函数。这是为检测 open 系统调用而设计的。参数 1 是打开文件的路径,参数 2 是标志位,参数 3 是打开模式。

● int timeout(struct timeval * start);

检查在等待队列中的任务是否超时。参数是任务开始的时间。

● void timecheck(void * noarg);

计时函数。每隔 1000 ticks,它将发送一个唤醒信号。

● create __ proc __ entry(AUDITDEV __ NAME,S __ IRUGO|S __ IWUSR,NULL);

在/proc 下注册一个文件的标准函数。

● copy __ to __ user(filebuffer,read __ position,bytes __ to __ write);

数据从内核空间到用户空间的标准函数。

● spin __ lock __ irqsave(&spin __ lock,flags);

Spinlock 机制中的“上锁”函数。

- spin __ unlock __ irqrestore(&spin __ lock,flags);

Spinlock 机制中的“解锁”函数。

3. 数据分析

(1) 实现机理

数据分析模块从数据采集模块的临时缓冲区中读入审计数据,并根据规则库里定义的规则对这些审计数据进行分析,看看是否有告警或可疑事件发生。它将用户的审计事件分为五种告警级别:严重警告、优先警告、警告、一般信息和无关信息。

对审计数据的分析分为两种模式,一种是基于内核的,另一种是基于目标的,用户只能选择其一。基于内核的模式是属于较低级的分析,它只是从所有的审计记录中将用户所关心的系统调用的审计数据筛选出来,然后由通信模块传送给上级 Agent;而基于目标的模式要高级一些,用户可以根据主机的情况来定义详细的规则库,在对审计数据进行分析的时候,如果有事件和规则库中定义的事件相匹配,则记录下该事件,再由通信模块将数据传送给上级 Agent,否则系统则认为这个事件是冗余的数据,不予处理。为了不增加主机的负载,我们没有对审计数据进行综合的分析和处理,只是采用模式匹配的方法来对规则库中定义的规则进行匹配,在分析 Agent 里提供了数据挖掘、统计分析等算法来对各个主机 Agent 传送来的数据进行综合的分析。

下面将仔细介绍一下规则库的配置。

在基于内核的模式下,规则库的配置非常简单,用户只要定义好所关心的系统调用就可以了。在我们的系统中,可以同时对 34 种系统调用进行审计,这些系统调用都是和系统安全紧密相关的。

```
open = 0
creat = 0
execve = 1
exit = 0
mkdir = 0
unlink = 0
mknod = 0
rmdir = 0
chown = 1
lchown = 0
chown32 = 0
lchown32 = 0
chmod = 0
symlink = 1
link = 0
rename = 0
reboot = 0
truncate = 0
```

```
chroot = 0
setuid = 1
setreuid = 1
setresuid = 1
setuid32 = 1
setreuid32 = 1
setresuid32 = 1
setgid = 1
setregid = 1
setresgid = 1
setgid32 = 0
setregid32 = 0
setresgid32 = 0
truncate64 = 0
socketcall = 1
create __ module = 0
```

在上述系统调用中，1 表示保留该系统调用的审计信息，0 表示不保留，用户不关心该系统调用，认为它不存在系统安全的问题。

基于目标的审计是对所收集的事件进行高级的控制。它将事件分为 10 组，每组有相应的过滤条件：

- 打开或建立一个文件或目录。
- 以只读方式打开一个文件。
- 写一个文件或目录，或创建一个文件或目录。
- 删除一个文件或目录。
- 开始或停止一个文件的执行。
- 修改系统、文件或目录的属性。
- 改变用户或组的 ID。
- 打开一个网络连接。
- 接受一个网络连接。
- 管理员事件。

对创建的每个目标，都可以给它们指定一个事件的安全级别。一共分为 5 级：严重警告、优先警告、警告、一般信息和无关信息。安全级别的使用可以让用户很快地识别事件的严重程度。

以下是对每个目标的过滤条件的介绍：

1）针对具体事件的过滤，每个事件都包含特定的要交互的核心信息。例如，“打开或创建一个文件或记录组”：文件或目录的名字作为核心信息；“开始或停止程序的执行组”：程序的名字将作为核心信息。事件的匹配有三种方式：部分匹配、完全匹配和特定表达式。部分匹配就是查找核心信息里是否包含指定的字符串。例如：在“读、写或创建一个文件或目录”里指定部分匹配字符串“pass”，那么下面列出的将是符合条件的文件名：

```
/etc/passwd
/usr/lib/passfilt.so
/home/red/Khyber _ pass.txt
```

完全匹配是准确地匹配指定的字符串。例如:如果匹配的字符串是 /etc/password,那么 /etc/passwd 则匹配成功,但是文件/etc/passwd.backup 匹配错误。

特定表达式匹配是一种较高级的匹配,它提供了一种可扩展的方式。例如".*[Pp]ass(word|wd).*",则/etc/passwd 和 /tmp/PasswordFile 匹配成功,而/etc/PASSWORD 和 /home/red/PaSsWoRd.txt 匹配失败。

2) 对用户的过滤,可以选择特定的一定数量的用户监控该用户的某些操作,也可以不监控某些用户,认为这些用户的某个操作是可信的。如果不指定用户的话,则对所有用户都要监控。

3) 根据返回值监控,每个时间的返回值是成功或失败,可以按照事件的返回值来对事件进行审计。

4) 对一些特定事件的过滤,有些事件包括 open()和 socketcall(),允许指定额外的过滤条件,以提供更灵活的搜索标准。

a. open(),open 事件提供了对打开标志的过滤,这些标志按照特定的表达式格式来指定,包括:O _ WRONLY,O _ RDWR,O _ RDONLY,O _ CREAT,O _ EXCL,O _ NOCTTY,O _ TRUNC,O _ APPEND,O _ NONBLOCK,O _ SYNC,O _ NOFOLLOW,O _ DIRECTORY,O _ LARGEFILE

例如下面的标志是以逻辑"或"来指定目标的,表示"每当一个文件以写的模式打开的时候检测":

```
open(O _ WRONLY|O _ RDWR|O _ CREAT|O _ TRUNC|O _ APPEND)
```

而下面的 3 个例子是来指定"读或写":

```
open(.*)
open(O _ .*)
open(O _ WRONLY|O _ RDWR|O _ RDONLY|O _ CREAT|O _ EXCL|O _ NOCTTY|O _
TRUNC|O _ APPEND|O _ NONBLOCK|O _ SYNC|O _ NOFOLLOW|O _ DIRECTORY|O _
LARGEFILE)
```

b. socketcall(),Linux 内核使用 socketcall 系统调用来为 connect、accept 以及一些相关的系统调用服务。为了监控 connect 和 accept 系统调用,我们必须在目标里指定系统调用的类型。

例如:socketcall(ACCEPT)仅仅监控 accept()调用。Socketcall(CONNECT)仅仅监控 connect()调用。Socketcall(.*)或 socketcall(CONNECT|ACCEPT)同时监控 accept 和 connect。

规则库的配置是由中心 Agent 通过图形界面来完成的,每一类事件对应不同的系统调用,每个主机 Agent 都对应一个配置文件。我们可以看一个例子:

```
criticality=4
```

```
event = open( . * ), creat, mkdir, mknod, link, symlink
return = Success      user! = root      match = ^/etc/shadow $
```

上述规则表示发生 open(. *), creat, mkdir, mknod, link, symlink 等系统调用时，如果返回值是 success，而所执行的文件里包含/etc/shadow，但用户不是超级用户时，其警告级别为 4，即严重警告。

另外，审计数据分析之后，系统将按一定的格式将分析结果写入日志文件中，或直接发送到上级 Agent，不同类型的系统调用的审计数据不同，所以它们的数据格式也不同。

(2) 主要数据结构和函数

1) 重要的数据结构，因为不同类型的系统调用事件所记录的信息也不同，所以我们在实现时，对系统调用进行了分类，定义了如下数据结构：

- 输入输出事件，如文件打开，读，写等，包含的信息有头信息，返回值信息，进程信息，文件路径信息，当前目录信息，文件属性信息。

```
typedef struct
{
header__token        t__header;
return__token        t__return;
process__token       t__process;
path__token              t__path;
path__token              t__pwd;      // Working directory
attributes__token    t__attributes;
} io__class;
```

- 改变文件模式的事件，如 chmod 等，包含的信息有：头信息，返回值信息，进程信息，文件目录信息，当前目录信息，新所有者信息。

```
typedef struct
{
header__token        t__header;
return__token        t__return;
process__token       t__process;
path__token              t__path;
path__token              t__pwd;    // Working directory
owner__token             t__owner;
} ch__class;
```

- 执行类事件，如 execve 等，包含的信息有：头信息，返回值信息，进程信息，文件路径信息，当前目录信息，参数信息。

```
typedef struct
{
header__token        t__header;
return__token        t__return;
```

```
process __ token        t __ process;
path __ token                t __ path;
path __ token                t __ pwd;   // Working directory
execargs __ token            t __ execargs;
// environment variables too?
| ex __ class;
```

- 进程控制类事件,包含的信息有:头信息,返回值信息,进程信息。

```
typedef struct
{
header __ token        t __ header;
return __ token        t __ return;
process __ token       t __ process;
| pc __ class;
```

- 复制文件类的事件,包含的信息有:头信息,返回值信息,进程信息,原文件路径,当前目录,目的文件路径。

```
typedef struct
|
header __ token        t __ header;
return __ token        t __ return;
process __ token       t __ process;
path __ token                t __ sourcepath;
path __ token                t __ pwd;   // Working directory
path __ token                t __ destpath;
| cp __ class;
```

- Setuid 类事件,包含的信息有:头信息,返回值信息,进程信息,目标用户或组信息。

```
typedef struct
{
header __ token        t __ header;
return __ token        t __ return;
process __ token       t __ process;
target __ token        t __ target;   // 目标用户 ID 或组 ID.
| su __ class;
```

- 网络连接事件,包含的信息有:头信息,返回值信息,进程信息,连接信息。

```
typedef struct
{
header __ token        t __ header;
return __ token        t __ return;
process __ token       t __ process;
```

```
int                          syscall;      /* 哪一类网络连接事件? 如 connect(),accept()
                                             listen() */
connection __ token    t __ connection;
| nt __ class;
```

- 管理员事件,包含的信息有:头信息,返回值信息,进程信息,路径信息。

```
typedef struct
}
header __ token        t __ header;
return __ token        t __ return;
process __ token       t __ process;
path __ token                t __ name;     // 加载/删除的模块名
| ad __ class;
```

- 规则节点的数据结构。

```
struct __ node
{
int event __ number;   // 规则 ID
int criticality;   // 严重程度
int returncode;        // 返回值
int excludeflag;       // 是否包括给定的用户
char username[MAX __ USERREG];   // 用户
char path[PATH __ MAX];          //路径
char options[MAX __ OPTIONS];    //可选项
struct __ node * next;
};
typedef struct __ node Node;
```

2) 重要的函数说明。

- int read __ config __ file():读取规则库中的规则,将所有规则组成一个队列,当从数据采集模块接收到审计信息后,将根据规则队列来判断这个事件是否是告警信息。
- Node * CheckObjective(int eventid,const char * username,const char * searchterm,char * options,int returncode):判断当前事件是否告警,传入的参数为当前事件的信息,它将根据规则队列来判断,返回值为匹配的规则节点。
- int process __ io __ class　　(header __ token *);
  int process __ ex __ class　　(header __ token *);
  int process __ pc __ class　　(header __ token *);
  int process __ ch __ class　　(header __ token *);
  int process __ cp __ class　　(header __ token *);
  int process __ su __ class　　(header __ token *);
  int process __ nt __ class　　(header __ token *);

由于不同的系统调用类型所需要记录的信息不同，所以它们分别用来处理不同的类型，首先将事件的信息按一定的格式记录下来，然后调用 checkObjective()来判断当前事件是否存在于规则库中，是否属于告警事件。

- void sig __ handler(int signal __ id)，用来处理不同的信号事件，包括从数据采集模块的缓冲区中读取审计事件数据，停止和重启主机 Agent 时所需要做的工作。

6.3 分析 Agent 的设计和实现

分析 Agent 在整个系统中处于中间层，它对各个主机 Agent 发送来的数据进行综合分析，以便检测到涉及多台主机与网络有关的入侵行为；分析 Agent 注重于高层次的分析方法，综合运用概率统计技术、数据挖掘技术以及神经网络技术，进行特征提取和模式匹配，不断学习，动态扩充入侵模式库，动态更新系统正常行为轨迹，以便发现异常入侵，充分发挥分析 Agent 的检测能力。在一个分布式的入侵检测系统中会存在多个不同的分析 Agent，每个分析 Agent 所采用的检测方法也不一定相同。即在一个分析 Agent 中也可以同时采用几种检测方法，对相同的数据使用不同的检测方法进行分析，对各自的检测结果进行比较，可以提高检测准确度，也可以完善不同的检测方法。

6.3.1 分析 Agent 的结构

分析 Agent 的结构如图 6-3 所示。

数据存储模块接收来自主机 Agent 的数据，并将数据存入数据库中，以便于以后的分析。在前面已经详细介绍了主机 Agent 的结构和具体实现，由主机 Agent 传送来的数据是已经过处理的数据，这样大大减少了网络数据的传输，降低了网络的负载。构造专用的存储系统可以提高反应速度，但那是一个艰难的过程。我们选择现有的数据库系统来建立存储系统。现有的数据库系统都支持 SQL 查询，它对搜索的封装使得分析系统可以使用通用的接口。

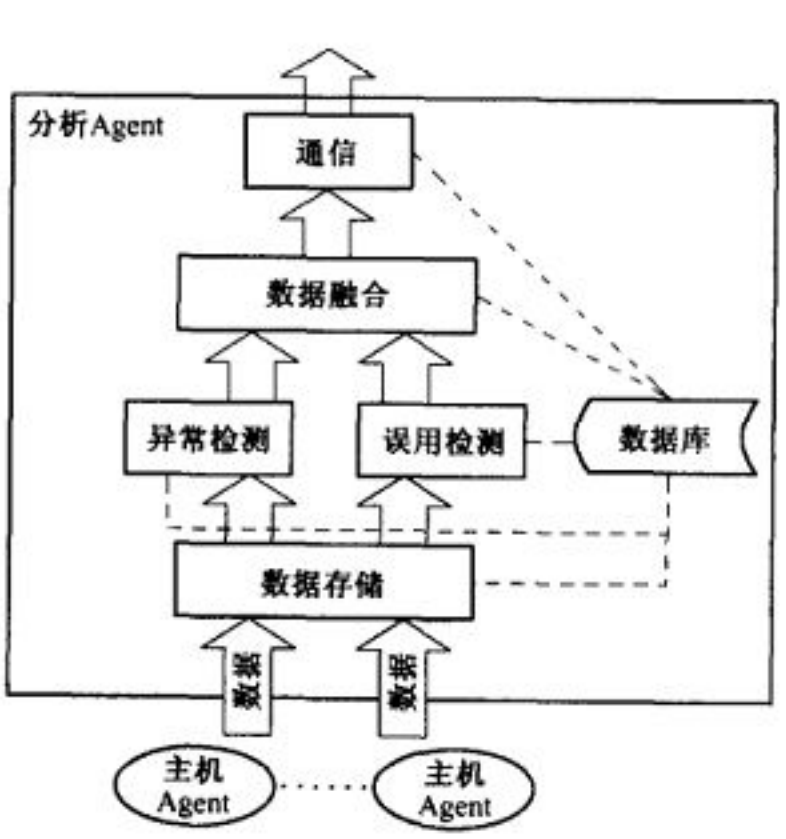

图 6-3 分析 Agent 的结构

根据入侵检测系统的规模，存储系统也有很大的区别，它可能是一个单一的数据文件，也可能是一个数据库系统，对于大规模的入侵检测系统也有可能配置一个数据仓库作为其存储系统。

对数据的分析我们采用了误用检测和异常检测相结合的方式，由于系统中存在多个分析 Agent，所以不同的分析 Agent 所采用的数据分析方法也不尽相同。目前误用检测技术有专家系统、模型推理、状态转换分析算法等，异常检测有统计分析、神经网络、数据挖掘、遗传和免疫等算法。在后面将针对主机审计数据的特点重点讲解几种常用的算法。

对数据进行分析之后，需要将不同算法的分析结果进行融合，同时将主机 Agent 的分析结果

和分析 Agent 的分析结果进行融合，存入到数据库中并通过通信模块传送给中心 Agent。

6.3.2　具体实现

分析 Agent 是完全独立于任何操作系统的，它所分析的数据来自各个不同的主机 Agent。在分析 Agent 中最主要的部分就是对数据采用不同的算法进行分析，对算法选择的好坏直接影响到入侵检测的准确率，所以在选择算法时一定要慎重地分析和考虑。我们通过对各种分析方法进行分析之后，决定采用以下算法，在某个分析 Agent 中可能会采用其中的一种或多种算法。

1. 数据的格式

分析 Agent 所分析的数据来源是主机 Agent，所有主机 Agent 发送来的数据都是统一的标准的格式，不需要再重新处理，这非常有利于分析 Agent 进行分析，节省了很多麻烦。下面看一下主机 Agent 所提供的数据的内容。

- criticality，time，details：事件的危险程度、事件发生的时间、事件的详细描述。
- event：event：发生的事件的 ID。
- user：realUserId，realGroupId，effectiveUserId，effectiveGroupId：事件的执行者，包括实际用户、实际组、有效用户、有效组。
- target：targetUserId：目标用户（系统调用 setuid 所设定的用户）。
- process：processId，processName：进程的信息，包括进程 ID、进程的名字。
- arguments：arguments：系统调用的参数。
- path：file：系统调用所执行的文件。
- attributes：attribute：文件属性。
- destpath：destinationFile：目标文件。
- owner：newOwner，newGroupOwner：文件的新的所有者，包括新户主和新组。
- socket：sourceIp，destIp，sourcePort，destPort：如果是网络连接的系统调用，则包含源 IP、目的 IP、源端口、目的端口。
- return：returnCode：系统调用的返回值。

可以看到，主机 Agent 提供了非常详细、丰富的数据用来分析。

2. 分析算法

对数据的分析是基于以上格式记录的，主要采用统计分析方法和数据挖掘算法，任务是发现系统活动的统计规律和模式。例如，在短时间内，某个账号多次登录失败、某用户多次企图访问某些关键文件失败、某用户花大量的时间浏览文件系统并查看系统信息、深夜登录系统、突然发生来自以前从未有过的 IP 地址的访问、网络连接突然增多等。

（1）概率统计方法

概率统计方法是基于行为的入侵检测中应用最早也是最多的一种方法。首先，检测系统根据用户对象的动作为每个用户建立一个用户特征表，通过比较当前特征与已存储定型的历史特征，从而判断是否是异常行为。用户特征表需要根据审计记录情况不断地加以更新。用于描述特征的变量类型有：操作密度（度量操作执行的速率，常用于检测通过长时间平均觉察不到的异常行为）、审计记录分布（度量在最新记录中所有操作类型的分布）、范畴尺度（度量在一定动作范畴内特定操作的分布情况）、数值尺度（度量产生数值结果的操作，如 CPU 使用量，

I/O 使用量)。

这些变量所记录的具体操作包括:CPU 的使用,I/O 的使用,使用地点及时间,邮件使用,编辑器使用,编译器使用,所创建、删除、访问或改变的目录及文件,网络上活动等。

在 SRI/CSL 的入侵检测专家系统(IDES)中给出了一个特征简表的结构:

<变量名,行为描述,例外情况,资源使用,时间周期,变量类型,门限值,主体,客体,值>

其中的变量名、主体、客体惟一确定了每一个特征简表,特征值由系统根据审计数据周期性地产生。这个特征值是所有有悖于用户特征的异常程度值的函数。如果假设 S1,S2,...,Sn 分别是用于描述特征的变量 M1,M2,...,Mn 的异常程度值,Si 值越大说明异常程度越大。则这个特征值可以用所有 Si 值的加权平方和来表示:

M=a1s12+a2s22+...+ansn2,ai>0,其中 ai 表示每一特征的权重。

如果选用标准偏差作为判别准则,则标准偏差:$\sigma=\sqrt{M/(n-1)-\mu 2}$,其中均值 $\mu=M/n$。

如果某 S 值超出了 $\mu\pm d\sigma$,就认为出现了异常。

常用的统计概率模型有操作模型、均值和方差模型、时间序列模型等。操作模型假设异常可通过测量结果与一些固定指标相比较得到,固定指标可以根据经验值或一段时间内的统计平均值得到,比如在一定时间内多次失败的登录很可能是口令尝试攻击。时间序列分析是将事件计数与资源耗用根据时间排成序列。如果一个新事件发生的概率较低,则该事件可能是入侵。通过对审计事件分析而发现的规律有些是系统正常活动表现出的特征,有些则是异常特征,这些异常特征预示着可能的入侵。检测系统必须通过学习来判断哪些规律是正常的,哪些是异常的。

这种方法的优越性在于能应用成熟的概率统计理论。但也有一些不足之处,如:统计检测对事件发生的次序不敏感,也就是说,完全依靠统计理论可能漏检那些利用彼此关联事件的入侵行为。其次,定义是否入侵的判断阈值也比较困难。阈值太低则漏检率提高,阈值太高则误检率提高。另外就是不能完全依靠自学获得知识,在某些情况下,必须有安全专家的参与才能获得某种规律是否预示着入侵的知识。如登录高峰期发生了转移可能是由于公司的作息时间发生了更改;如突然出现了来自新 IP 地址的登录或连接,可能是由于管理员为新员工购买了新机器分配了新的 IP 地址,检测系统不可能预知这些信息,当其发出错误的警报时由管理员更正,从而可能获得新的知识。另外,对于已知的入侵方式产生异常现象的知识,也可由安全专家手工输入。

(2) 数据挖掘方法

在 IDS 中应用数据挖掘技术提取用户行为特征,数据挖掘(Data Mining,简称 DM)是一种决策支持过程,它主要基于人工智能、机器学习和统计等技术,它能高度自动化地分析原有的数据,做出归纳性的推理,从中挖掘出潜在的模式,预测出用户的行为。DM 的技术基础是人工智能,它利用了人工智能的一些已经成熟的算法和技术,例如:人工神经网络、遗传算法、决策树、邻近搜索算法、规则推理、模糊逻辑等。DM 系统利用的技术越多,得出的结果精确性就越高。这主要取决于问题的类型以及数据的类型和规模。

在我们的系统中主要采用了关联分析算法。关联分析的目的就是挖掘出隐藏在数据间的相互关系。它就是给定一组 Item 和一个记录集合,通过分析记录集合推导出 Item 间的相关性。在网络安全系统中,可以用关联分析来找出入侵者的各种入侵行为之间的相关性。

另外我们也可以参照序列模式,序列模式分析和关联分析相似,其目的也是为了挖掘数据

间的联系,但序列模式分析的侧重点在于分析数据间的前后关系。黑客的许多入侵行为都是有先后关系的,有的行为必须发生在其他的行为之后。例如:黑客在真正实施攻击时,一般要对系统的端口进行扫描等。

1) 关联规则的基本概念。设 I = {i1,i2,...,im} 是二进制文字的集合,其中的元素称为项(item)。记 D 为交易(transaction)T 的集合,这里交易 T 是项的集合,并且 T⊆I。对应每一个交易有惟一的标识,如交易号,记作 TID。设 X 是一个 I 中项的集合,如果 X⊆T,那么称交易 T 包含 X。

一个关联规则是形如 X⇒Y 的蕴涵式,这里 X⊂I,Y⊂I,并且 X∩Y = Φ。规则 X⇒Y 在交易数据库 D 中的支持度(support)是交易集中包含 X 和 Y 的交易数与所有交易数之比,记为 support(X⇒Y),即:

$$\text{support}(X \Rightarrow Y) = |\{T: X \cup Y \subseteq T, T \in D\}| / |D|$$

规则 X⇒Y 在交易集中的可信度(confidence)是指包含 X 和 Y 的交易数与包含 X 的交易数之比,记为 confidence(X⇒Y),即:

$$\text{confidence}(X \Rightarrow Y) = |\{T: X \cup Y \subseteq T, T \in D\}| / |\{T: X \subseteq T, T \in D\}|$$

给定一个交易集 D,挖掘关联规则问题就是产生支持度和可信度分别大于用户给定的最小支持度(minsupp)和最小可信度(minconf)的关联规则。

2) 关联规则挖掘的算法。Agrawal 等在 1993 年设计了一个基本算法,提出了挖掘关联规则的一个重要方法,这是一个基于两阶段频集思想的方法,关联规则挖掘算法的设计可以分解为两个子问题:

第 1 步:找到所有支持度大于最小支持度的项集(Itemset),这些项集称为频集(Frequent Itemset)。

第 2 步:使用第 1 步找到的频集产生期望的规则。

这里的第 2 步相对简单一点。如给定了一个频集 Y = I1I2...Ik,k≥2,Ij∈I,产生只包含集合{I1,I2,...,Ik}中的项的所有规则(最多 k 条),一旦这些规则被生成,那么只有那些大于用户给定的最小可信度的规则才被留下来。

为了生成所有频集,使用了递推的方法。其核心思想如下:

```
L1 = {large 1 - itemsets};
for(k = 2;Lk - 1((;k + + )do begin
Ck = apriori - gen(Lk - 1);      //新的候选集
for all transactions t∈D do begin
Ct = subset(Ck,t);      //事务 t 中包含的候选集
for all candidates c∈Ct do
c.count ++ ;
end
Lk = {c(Ck|c.count(minsup}
end
Answer = ∪kLk;
```

首先产生频繁 1 - 项集 L1,然后是频集 2 - 项集 L2,直到得到某个 r 值使得 Lr 为空,这时

算法停止。这里在第 k 次循环中,过程先产生候选 k－项集的集合 Ck,Ck 中的每一个项集是对两个只有一个项不同的属于 Lk－1 的频集做一个(k－2)－连接来产生的。Ck 中的项集是用来产生频集的候选集,最后的频集 Lk 必须是 Ck 的一个子集。Ck 中的每个元素需在交易数据库中进行验证来决定其是否加入 Lk,这里的验证过程是算法性能的一个瓶颈。这个方法要求多次扫描可能很大的交易数据库,即如果频集最多包含 10 个项,那么就需要扫描交易数据库 10 遍,这需要很大的 I/O 负载。

Agrawal 等引入了修剪技术(Pruning)来减小候选集 Ck 的大小,由此可以显著地改进生成所有频集算法的性能。算法中引入的修剪策略基于这样一个性质:一个项集是频集当且仅当它的所有子集都是频集。那么,如果 Ck 中某个候选项集有一个(k－1)子集不属于 Lk－1,则这个项集可以被修剪掉不再被考虑,这个修剪过程可以降低计算所有候选集的支持度的代价。

3)提取关联规则。我们把主机 Agent 传送来的数据作为我们挖掘关联规则的数据源,由于主机 Agent 提供的信息很详细,我们不可能把事件的所有信息都用来进行挖掘。在分析时我们将所有事件分为两大类,一类是有关文件操作的事件,另一类是有关网络连接的事件。

有关文件操作的事件的数据项有:agent,criticality,user,event,process,path,succ/fail,它们分别为 agent 的名字,事件级别,用户,事件名称,进程名称,存取文件,返回值。

有关网络连接事件的数据项有:agent,criticality,user,event,process,sourIP,sourPort,destIP,destPort,succ/fail,它们分别为 agent 的名字,事件级别,用户,事件名称,进程名称,源 IP,源端口,目的 IP,目的端口,返回值。

在用关联算法提取规则时,我们是将前一周的历史数据作为依据,利用分析的结果来建立正常模式库。我们认为这一周的数据是没有任何攻击的,所建立的正常模式库描述了用户的正常行为特征。正常模式库建好后,当有新的事件发生时,将新事件与正常模式库中的规则相匹配,如果在模式库中存在这个事件,则说明这个事件是正常的,否则认为该事件存在攻击。

建立用户的正常模式库并不是一个很简单的事情,需要有大量的数据作为数据源,这样才能更准确地挖掘到用户的正常行为规则。但大量的数据又可能会导致许多误用的冗余的规则,这些规则并不是我们所需要的,所以应该剔除掉。我们认为,任何攻击都是和某个特定的用户相关的,所以在提取到的规则中必须包含用户属性,我们将用户属性作为轴属性。任何与非轴属性之间的关系对我们来说都没有太多的意义,可以抛弃掉。另外可以通过设定合理的最小支持度和最小可信度来控制有效规则产生的数量。

由于用户的行为不可能是一成不变的,所以必须及时地更新正常模式库,来适应各种不同情况的发生。

3. 数据融合

由于分析 Agent 可能采取不同的分析算法对审计数据进行分析,所以有必要将不同的分析结果进行融合,另外要将主机 Agent 分析处理后的结果和分析 Agent 分析处理后的结果进行融合,最后形成统一的格式传送到中心 Agent。在这里,我们并没有对数据融合进行深入的分析,只是简单地把处理后的结果融合在一起,除去重复、冗余的数据。

6.4　中心 Agent 的设计和实现

中心 Agent 位于整个系统的最高层，它负责监控系统中的所有 Agent，对 Agent 进行配置管理，显示告警信息并对告警事件进行相应的处理。它也是系统跟用户的接口部分，相当于一个控制台。

6.4.1　中心 Agent 的结构

中心 Agent 的结构如图 6-4 所示。

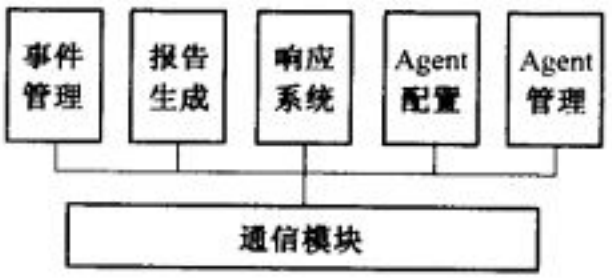

图 6-4　中心 Agent 的结构

1. 事件管理

事件管理模块管理的事件是由主机 Agent 和分析 Agent 所产生的告警事件，这些事件包括入侵警报、可疑行为等。事件管理要解决如何有效地显示以及如何与用户交互的问题。系统将显示该事件的所有详细信息。

我们对各种不同的告警事件进行分类显示，按不同的颜色区别不同威胁程度的攻击，按时间顺序列出攻击的详细情况，按相同 IP 地址将攻击分类等等。

对于事件管理，仅仅列出攻击本身的信息是不够的。对于每一个攻击，应该给管理员提供更多的信息，比如该攻击的解决方法，在 Internet 上的相关连接等。对于管理员来说，标记已经分析的攻击也是一个非常实用的功能。

有两种类型的报告：事件检测报告和总结报告。事件检测报告提供关于检测的低层次的详细信息，而总结报告则帮助分析员了解攻击者的趋势，看哪里需要进行重点保护。

2. 报告生成

(1) 事件检测报告

事件检测报告通常以事件为单位产生，或以一天的总结报告的形式产生。它们常以电子邮件的方式发送，而且入侵检测系统应该提供灵活的地址并能够进行 PGP 加密。分析员应有机会报告控制台上显示的每一个检测事件，而且可以通过点击鼠标接受每个检测事件。然后，系统自动产生报告，分析员在报告发送之前进行检查和注释。

(2) 周、月总结报告

管理部门通常对自己负责的站点是否正在受到攻击感兴趣。但是以事件为单位或以天为单位的报告需要花费很多时间来处理，而且无法帮助他观察宏观的形势。周或月总结报告解决了这个问题。通常，管理者的级别越高，向他发送报告的频率就应越低。

系统管理员通过对报告的分析，针对不同功能的 Agent 制定更好的策略。

3. Agent 的管理和配置

由于系统中存在多个不同级别、不同功能的 Agent，不同 Agent 所采取的分析方法也不同，所以有必要对这些 Agent 进行统一的管理和配置。我们要记录各个 Agent 的名字、IP 地址、通信端口以及所实现的功能等等，并能够停止和启动不同的 Agent。

对主机 Agent 而言，不同的主机 Agent 所使用的规则库也是不一样的，所以中心 Agent 需要对主机 Agent 的规则库进行重新配置，以适应各种不同的情况。

4. 响应系统

响应是入侵检测系统必须的一个部分，没有它入侵检测就失去了存在的价值。最简单的自动响应是自动通知。当检测到入侵发生时，入侵检测系统可以给管理员发 email 或寻呼。比较主动的响应是阻止正在进行的攻击，使得攻击者不能够继续访问。例如通过注入复位数据报来截断攻击者和目标主机之间的连接、通过更新配置防火墙来限制入侵者的访问等。更为主动的响应是探测到进攻时发起对攻击者的反击。但这个方法是非常危险的，它不但是非法的，也会影响网络上无辜的用户。这个方法如图 6-5 所示。

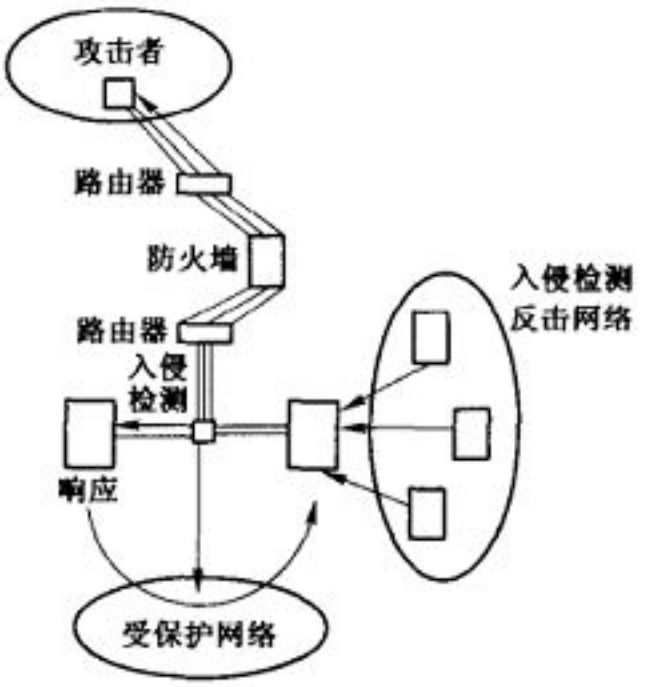

图 6-5 反击网络

自动响应是最便宜，最容易的响应方式，这种事故处理形式应广泛实行。只要实施适当，它也还是比较安全的。但其中存在两个问题：

第一个问题是，既然入侵检测系统有产生误报警的问题，我们就有可能错误地针对一个从未攻击我们的网络节点进行响应。就像那些攻击我们的人一样，我们也不太容易被发现，也就是说，这不会引起什么损害。由于 IP 地址欺骗和误报警可能引起的错误，也许有时不得不放弃利用自动响应的方式回击攻击者。

另一个问题是，如果攻击者判定我们的系统有自动响应，他可能会利用这一点来针对我们。例如，他可能与两个带自动响应入侵检测系统的网络节点建立起一个与 echo－chargen 等效的反馈环，再对那两个节点进行地址欺骗攻击。或者攻击者可从某公司的合作伙伴/客户/供应商的地址发出虚假攻击，使得防火墙把一个公司与另一个公司隔离开，这样两者之间就有了不能逾越的隔离界限。

6.4.2 具体实现

中心 Agent 各个部分的功能是完全通过图形化界面来实现的，这样有利于用户管理整个系统，便于用户操作。考虑到主机 Agent 的跨平台性，我们采用了 Java 语言进行界面的设计和各种功能的实现。

1. 事件管理

事件显示的主界面如图 6-6 所示，用来显示不同 Agent 的告警事件。界面的左侧是 Agent 列表，选择不同的 Agent 可以来查看发生在不同 Agent 上的告警和可疑事件。每个 Agent 的事件是按时间发生的先后次序来显示的，在主界面上显示事件的告警级别、发生时间和事件的详细描述。为了便于用户方便地查看每个事件的详细信息，我们采用了侦听机制，双击该事件即可弹出一个窗口来显示该事件的详细信息(如图 6-7 所示)。

为了保证入侵检测系统的实时性，系统采用了多线程机制来实时显示不同 Agent 的告警事件，以保证系统管理员及时发现这些告警事件，并采取相应的措施。

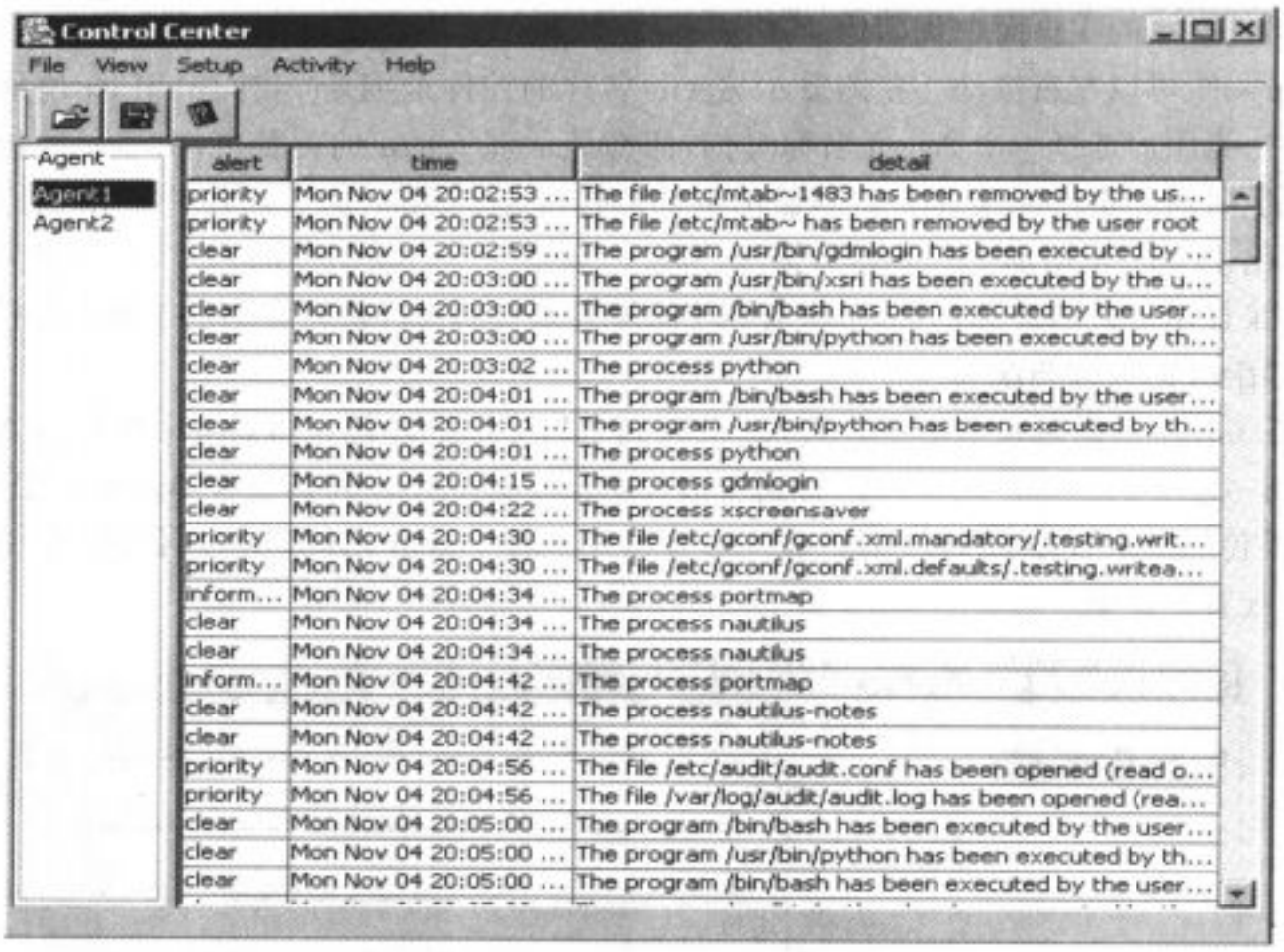

图 6-6　事件显示主界面

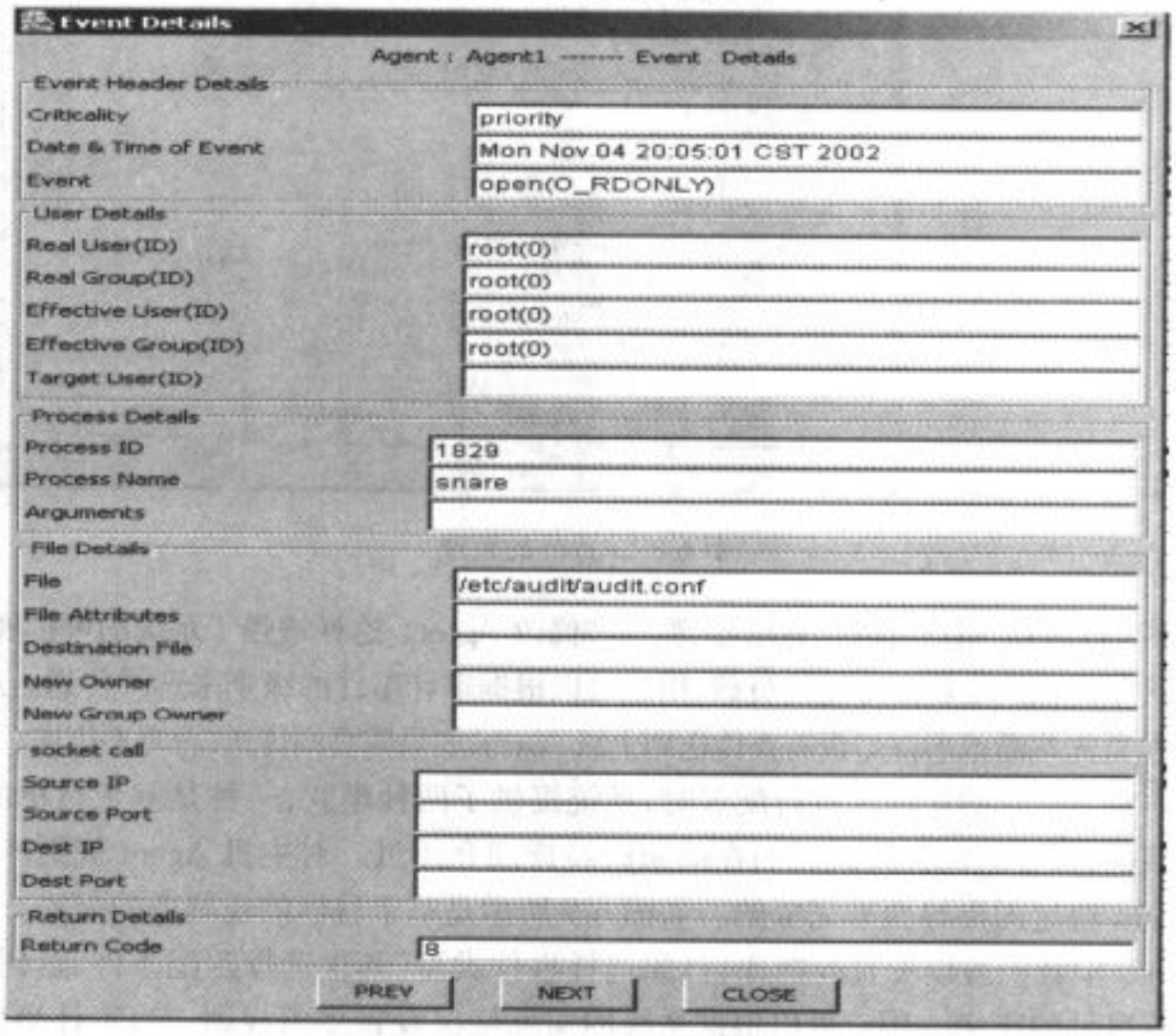

图 6-7　事件的详细信息

在事件显示的主页面中我们用 JSplitPane 将 Agent 列表和事件的显示分为左右两部分，中间的分隔线可以左右拉动。左侧显示 Agent 名称的控件是 JList，我们给 JList 增加了一个侦听事件，当用户选择一个 Agent 名称时，右侧将显示该 Agent 的告警事件，显示的信息是事件的级别、时间和事件的描述。右侧显示事件的控件是 JTable，为了给用户提供更详细的信息，我们给 JTable 也增加了一个侦听事件，当用户双击一个告警事件时，系统会弹出另一个新的窗口，来显示事件的详细信息，这样有利于系统管理员对事件进行详细的了解和分析。

2. Agent 的管理与配置

如图 6-8 所示是对系统中所有的 Agent 进行管理。图的左侧显示系统中所有的 Agent，右侧显示该 Agent 的一些信息。可以看到右边显示的是该 Agent 和其上级 Agent 的 IP 和端口以及需要传送的审计事件的级别。这些信息是用来使 Agent 之间进行通信的，存储于中心 Agent的配置文件里。

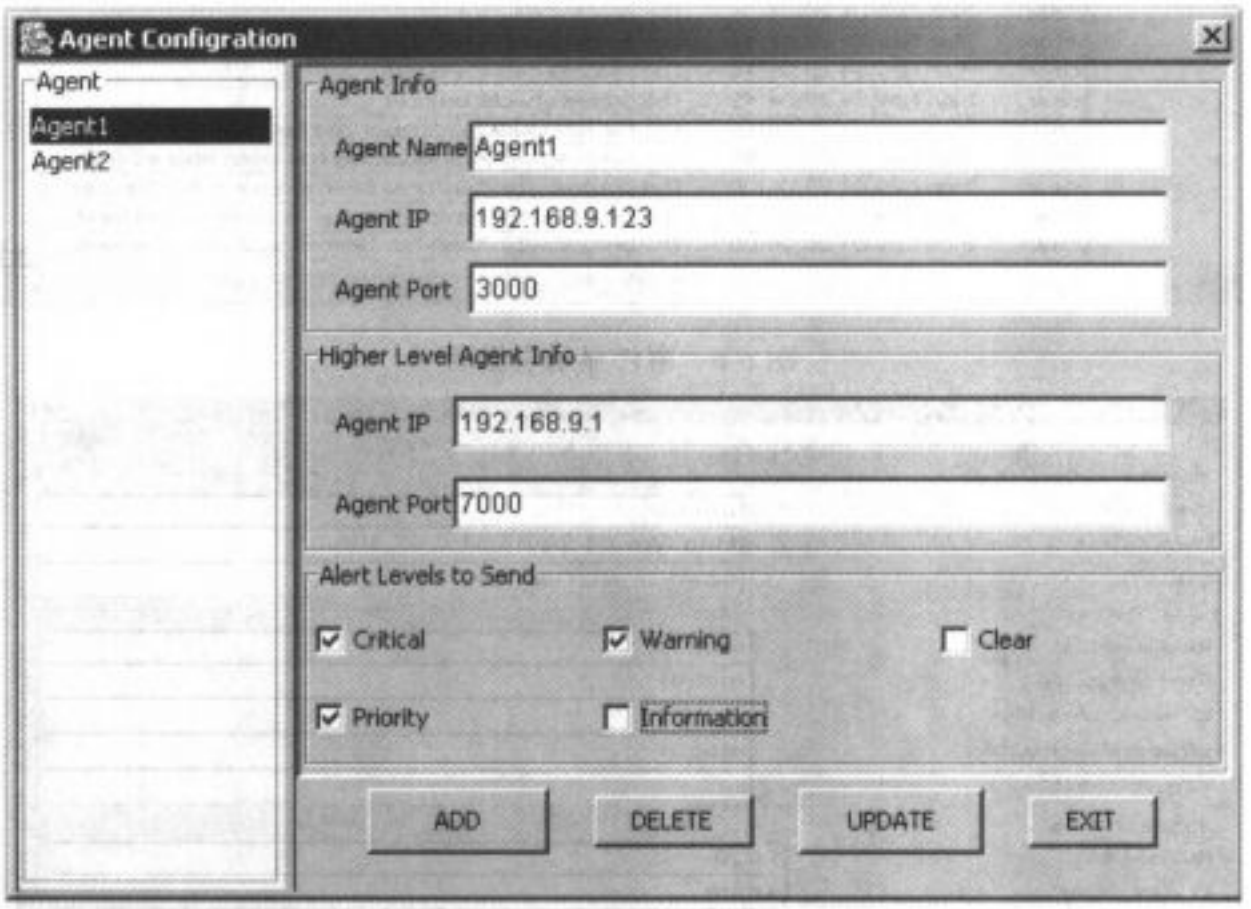

图 6-8 Agent 的管理

从图中可以看到，用户可以任意增加、删除和修改 Agent。这样增强了系统的可扩展性和灵活性。另外，为了降低网络数据传送的负载，用户可以根据审计事件的级别来选择传送。Agent 开始数据传送时，首先判断该事件是否需要传送到上级 Agent，如果需要则传送，否则不传送。

在对主机 Agent 的规则库进行配置时，系统提供了两种模式，一种是基于目标的，另一种是基于内核的，关于这两种模式我们在前面已经详细介绍过。对主机 Agent 规则库的配置是由中心 Agent 通过图形界面来完成的。如图 6-9 所示是基于目标的规则库的配置。

在图 6-9 中我们可以配置规则库，首先选择事件类型，其次选择匹配条件、返回值、用户，最后确定该事件的级别。用户可以根据系统的实际情况设置多条规则。当审计事件发生时，主机 Agent 根据规则库中定义的规则来匹配是否有告警事件发生。

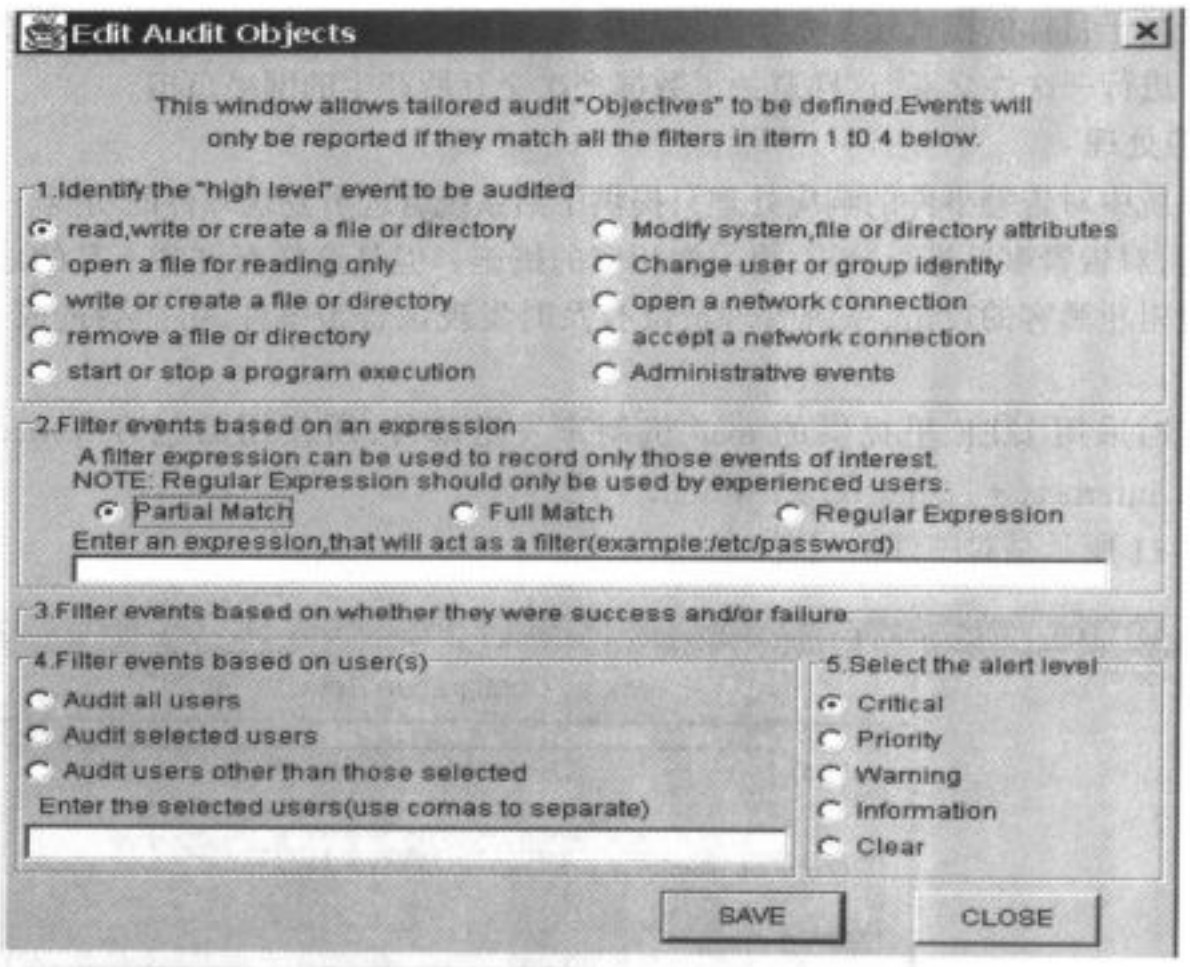

图 6-9　目标规则的配置

系统中提供了对 34 种与安全相关的系统调用事件的监控，如图 6-10 所示，用户可以选择监控其中的任何事件。这些事件包括资源存取和安全类系统调用、执行命令类和资源的创建和删除类的系统调用，基本上包含了所有的与系统安全相关的事件。

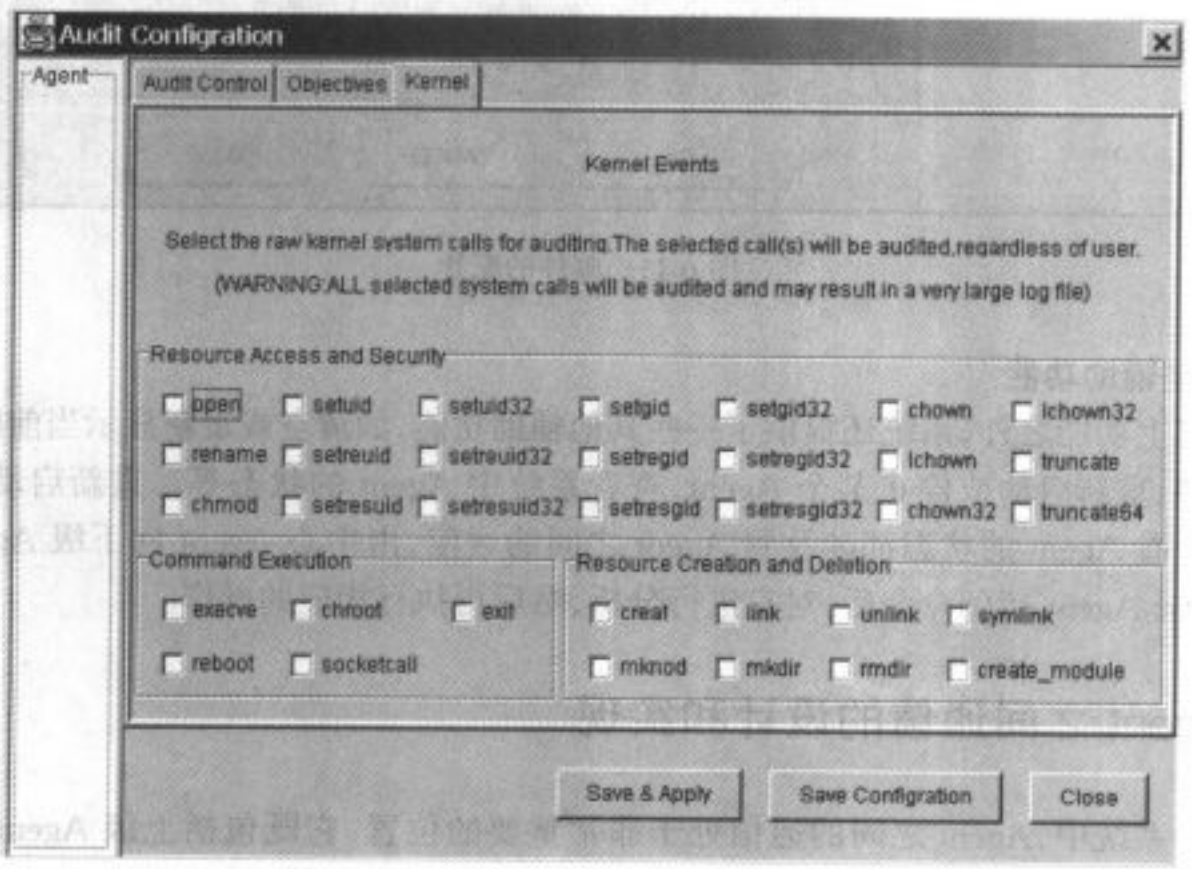

图 6-10　基于内核的配置

不论是基于目标的模式还是基于内核的模式，系统都会把配置信息存入一个文件中，然后与该 Agent 进行一次性交互，这样避免了数据多次交互所产生的网络负担。

3. 响应处理

目前系统中对告警事件的响应处理只提供了给系统管理员发送邮件的功能。系统管理员收到邮件后，对告警事件进行分析，再采取相应的措施。但是这样有许多不足的地方，例如可能没有及时阻止黑客的攻击，系统管理员没有及时发现该告警事件，可能会造成比较大的损失。

在实现时采用 J2EE 里提供的 mail 机制来发送邮件，所使用的包有“javax.mail.*”、“javax.mail.internet.*”、“javax.activation.*;”

如图 6-11 所示是对邮件的配置。

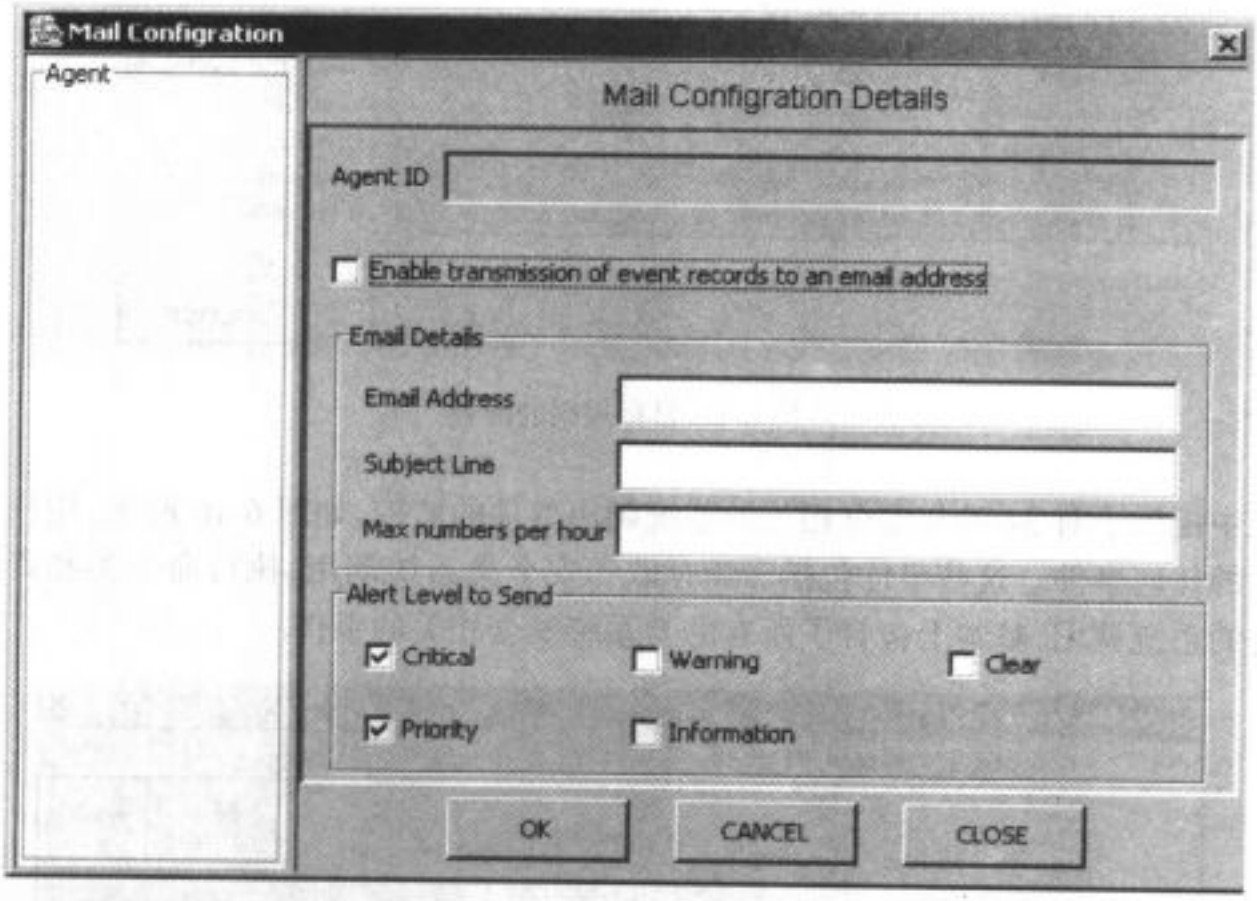

图 6-11 邮件的配置

4. 其他辅助功能

除了以上功能之外，系统还提供了一些其他辅助功能，如清空或重新显示当前屏幕上显示的告警事件、重新启动或停止某个 Agent，查看系统中 Agent 的状态等。重新启动或停止 Agent 以及查看 Agent 的状态都涉及到 Agent 之间的通信，由中心 Agent 向下级 Agent 发送控制命令，下级 Agent 接到命令后，对它进行分析，然后再执行相应的操作。

6.5 Agent 之间通信的设计和实现

在整个系统中，Agent 之间的通信处于非常重要的位置，它既包括上级 Agent 对下级 Agent 的控制，也包括下级 Agent 向上级 Agent 所传送的告警数据，以及中心 Agent 对所有 A-

gent 的配置。由于 Agent 之间传送的数据非常重要和敏感，所以在数据传送时应采取相应的加密措施，防止数据泄漏。我们采用的是较为简单和通用的字符流方式和 TCP/IP 协议。

我们将 Agent 之间的通信分为两大类：一是 Agent 之间告警信息的传送，主要是下级 Agent 将分析结果传送给上级 Agent；另一类是上级 Agent 对下级 Agent 的管理和控制。由于在我们的系统中分析 Agent 的数据来源是主机 Agent，而且是采用一定的算法来分析的，不需要太多的配置，所以我们只是实现了中心 Agent 对主机 Agent 的配置。

6.5.1　告警审计数据的传送

在通信模块中，SOCKET 通信是关键技术。由于主机 Agent 处于 Linux 主机上，故在通信中采用了 GNUC 的 SOCKET 通信。而分析 Agent 是不依赖于任何操作系统的，考虑到跨平台和通用性，在通信中采用 Java 的 SOCKET 通信。由于在设计时，考虑到这个系统要在被检测主机开机启动时就对它进行检测，所以通信双方都应该设计成为开机自启动的守护进程的模式。

1. 主机 Agent 端

它的主要任务是连接到分析 Agent，向分析 Agent 发送经过分析和处理的审计信息，因此这里涉及到三个方面的实现技术。首先是这个子模块存在(运行)的方式——守护进程机制。守护进程(Daemon)是运行在后台的一种特殊进程，它独立于控制终端并且周期性地执行某种任务或等待处理某些发生的事件。Linux 的大多数服务器就是用守护进程实现的。守护进程最重要的特性是后台运行，在这一点上与 DOS 下的常驻内存程序 TSR 相似。其次，守护进程必须与其运行前的环境隔离开来。这些环境包括未关闭的文件描述符，控制终端，会话和进程组，工作目录以及文件创建等。这些环境通常是守护进程从执行它的父进程(特别是 shell)中继承下来的。最后，守护进程的启动方式有其特殊之处。它可以在 Linux 系统启动时从启动脚本/etc/rc.d 中启动，可以由作业规划进程 crond 启动，还可以由用户终端(通常是 shell)执行。在实现中，采用守护进程的模式，在一定程度上降低了资源消耗，提高了系统性能，同时也满足了功能上的需求。

Linux 系统 SOCKET 通信的过程和步骤基本是与 Windows 类似的，也分为创建套接字，设置 IP 地址、端口号等属性，建立连接，传送数据，关闭连接等步骤。从实现的功能上分析，正如前面提到的那样，在创建套接字时，使用的是面向连接的数据流方式。由于只向一个分析 Agent 传送信息，因此在设置属性时，只需获得该分析 Agent 主机名或 IP 地址和端口号，这些都是在中心 Agent 里通过图形界面来配置好的。由于系统对 Linux 的检测是实时的，因此传输的数据量和速度也是应该考虑的问题。这里关键的问题是数据传送缓冲区的设置。考虑到功能上要对每一条审计日志进行简单分析处理后再进行传送，同时在实际运行的情况下，实时产生的审计日志经过分析处理后需要传送的数据量应该不会特别大，为了提高传送的效率，这里并没有采用构造可容纳多条审计日志数据的缓冲区，而仅是以一条审计日志数据作为每次传送的对象。

在这里还实现了一些辅助的功能。主机 Agent 对审计信息分析处理后对事件分为 5 级，为了减少数据传输量，提高性能，我们进行了优先级过滤，也就是通过获取配置信息中的优先级过滤信息，对每一条审计日志信息进行分析，对于符合的日志信息进行传送，而对于其他的

则被过滤掉。

2. 分析 Agent 端

它的主要任务是接受来自主机 Agent 的连接，并将传来的审计数据存放到文件或数据库中，以便分析 Agent 对这些数据进行二次分析和处理。由于有多个主机 Agent 通过不同的端口连接到分析 Agent 传送数据，为了保证效率，体现处理的并行性，在这里采用了 Java 中的多线程机制。对于每一个 Agent 传送数据，都对应一个线程，既提高了并发处理能力，同时也增强了扩展性。由于 Java 中对网络(SOCKET)通信封装得非常好，并且由于是服务器端，也不需要设置其他的属性，因此实现起来比较简单。

分析 Agent 和中心 Agent 之间告警信息的传送也是用 Java 来实现的，和分析 Agent 类似，所以在此不再赘述。

6.5.2 控制信息的传送

Agent 之间控制信息的传送主要包括上级 Agent 对下级 Agent 的控制。在我们的系统中实现了中心 Agent 对主机 Agent 和分析 Agent 的控制和管理。由于分析 Agent 的配置非常简单，所以在此只讲述中心 Agent 对主机 Agent 的控制和管理的实现。

中心 Agent 对主机 Agent 的控制包括对主机 Agent 的规则库的配置、重新启动或停止当前运行的主机 Agent、获得主机 Agent 的状态信息等。中心 Agent 相当于一个控制台，它给用户提供了一个图形界面对系统中的所有的 Agent 进行配置和管理。用户可以在中心 Agent 端查看任何 Agent 的配置和状态。

当用户要求查看某个主机 Agent 的配置时，中心 Agent 会与该主机 Agent 建立连接，发出请求，主机 Agent 会把其配置文件发送给中心 Agent，然后中心 Agent 将配置文件中的信息显示到图形界面中。如果用户需要对主机 Agent 的规则库的配置进行修改，则主机 Agent 将会把修改后的配置信息发送到主机 Agent，以更新主机 Agent 的配置。

由于主机 Agent 配置的修改会影响到主机 Agent 对审计信息的分析结果，所以在修改主机 Agent 的配置之后，系统应该重新启动该主机 Agent，按照新的配置来运行。

另外，中心 Agent 也可以直接向主机 Agent 发送控制命令(停止或重启主机 Agent)。当主机 Agent 接收到中心 Agent 发送来的信息后，它首先判断该信息是什么请求，是重启 Agent、停止 Agent、获得 Agent 的状态还是获得配置信息。主机 Agent 会根据不同的请求做出不同的处理。

控制信息的通信都是通过 Java 的 socket 通信机制来实现的。

6.6 练习题

一、选择题

1. Agent 是指一个在主机上执行特定安全监测任务的(　　)，是一个可独立运行的实体。

A. 硬件代理　　B. 软件代理

C. 进程　　D. 线程

2. 中心 Agent 着眼于整个网络，主要完成的功能有(　　)。

A. 对分析 Agent 不能判定的情况进行判定。
B. 对系统中的所有 Agent 进行统一的配置和管理。
C. 实时接收来自数据源的数据。
D. 显示告警信息并进行响应处理。

3. 主机 Agent 处于整个分布式系统的(　　)层。
A. 最低　　B. 中间
C. 上层　　D. 最高

4. 事件管理模块管理的事件来自于(　　)。
A. 主机 Agent　　B. 分析 Agent
C. 中心 Agent　　D. 控制台

5. 对数据进行分析之后,需要将不同算法的分析结果进行(　　)。
A. 比较　　B. 融合
C. 分析　　D. 处理

6. 不能作为 IDS 中数据采集模块的数据源的是(　　)。
A. 操作系统的审计日志　　B. 应用程序的运行日志
C. 网络数据包　　D. 系统中存放的应用程序

二、简答题

1. 简述传统入侵检测系统的局限性。
2. 简述在入侵检测中,利用 Agent 来采集和分析数据的特点。
3. 简述主机 Agent 的实现机理。
4. 简述数据融合的意义。
5. 简述中心 Agent 的主要作用。
6. 试述在系统中,控制信息是如何传送的?

第7章　Snort分析

本章导读：

本章对开放源代码的入侵检测软件Snort进行了详细的分析。首先介绍了Snort的安装与配置方法；接着介绍了Snort的使用方法；然后分析了Snort使用的规则；最后对Snort的总体结构进行了分析。

7.1　Snort的安装与配置

本节简要介绍Snort的基本情况、特点和系统组成，然后介绍Snort的安装与配置，包括其底层库的安装与配置。

7.1.1　Snort简介

Snort是一个用C语言编写的开放源代码软件，符合GPL(GNU General Public License)的要求，当前的最新版本是1.8，其作者为Martin Roesch。

1. 基本情况

Snort是一个跨平台、轻量级的网络入侵检测软件，实际上它是一个基于libpcap的网络数据包嗅探器和日志记录工具，可以用于入侵检测。从入侵检测分类上来看，Snort应该算是一个基于网络和误用的入侵检测软件。

Snort采用基于规则的网络信息搜索机制，对数据包进行内容的模式匹配，从中发现入侵和探测行为，例如：缓冲区溢出、端口扫描、CGI攻击和SMB探测等。Snort具有实时报警的能力，它的警报信息可以发往syslog、SMB(Server Message Block)或者单独的alert文件。可以通过命令行进行交互，并对可选的BPF(Berkeley Packet Filter)命令进行配置。

Snort安装在一台主机上对整个网络进行监视，其典型运行环境如图7-1所示。

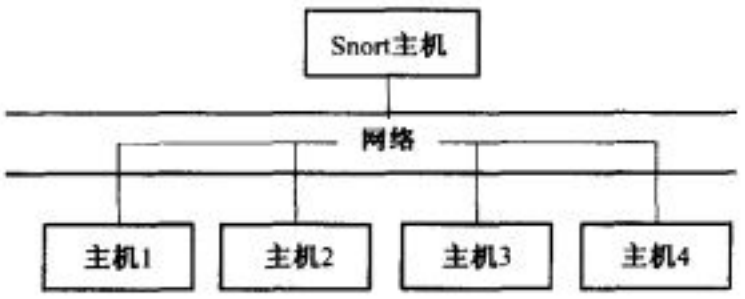

图7-1　Snort的典型运行环境

2. 特点

Snort具有跨平台的特点，它支持的操作系统非常广泛，包括Linux、OpenBSD、FreeBSD、NetBSD、Solaris、SunOS、HP-UX、AIX、IRIX、MacOS X Server、Win32(Win9x/NT/2000)等。而大多数商用入侵检测软件只能支持其中的一、两种操作系统，甚至需要特定的操作系统。

轻量级是指系统管理员可以轻易地将Snort安装到网络中去，可以在很短的时间内完成配置，可以很方便地集成网络安全的整体方案，使其成为网络安全体系的有机组成部分。

Snort基于规则的检测机制十分简单和灵活，它可以迅速地对新的入侵行为做出反应，可

以迅速填补网络中潜在的安全漏洞，例如当 IIS Showcode web exploits 在 Bugtraq mailing list 上公布不到几个小时，Snort 相应的规则就会发布。

Snort 在用于数据包嗅探器方面和 tcpdump 有一些类似，比如都使用 libpcap 和 BPF，但是 Snort 把主要重点放在数据包嗅探器的安全应用功能上，它具有 tcpdump 所没有的数据包负载监察功能，而且在收集数据的时候不进行主机名的查找，可以进行高效的数据包分析。另一个优点在于，它的输出形式比 tcpdump 要友好得多。

Snort 的规则表示比较简单，不过在处理 IP 分片重组方面功能不是很强。Snort 最直观的优点在于免费，因为大多数商用软件的代价都是非常高的，动辄需要成千上万元。此外，Snort 拥有众多的支持者，而且网上相关的资源也十分丰富。

3. 系统组成

Snort 由三个重要的子系统构成：数据包解码器、检测引擎、日志与报警系统。

(1) 数据包解码器

数据包解码器主要是对各种协议上的数据包进行解析、预处理，以便提交给检测引擎进行规则匹配。解码器运行在各种协议之上，从数据链路层到传输层，最后到应用层。因为当前网络中的数据流速度很快，如何保障较高的速度是解码器子系统中的一个重点。目前，Snort 解码器支持的协议包括 Ethernet、SLIP 和 raw(PPP)data-link 等。

(2) 检测引擎

Snort 用一个二维链表存储它的检测规则，其中一维称为规则头，另一维称为规则选项。规则头中放置的是一些公共的属性征，而规则选项中放置的是一些入侵特征。为了提高检测的速度，通常把最常用的源目的 IP 地址和端口信息放在规则头链表中，而把一些独特的检测标志放在规则选项链表中。规则匹配查找采用递归的方法进行，检测机制只针对当前已经建立的链表选项进行检测。当数据包满足一个规则时，就会触发相应的操作。Snort 的检测机制非常灵活，用户可以根据自己的需要很方便地在规则链表中添加所需要的规则模块。

(3) 日志和报警子系统

日志和报警子系统可以在运行 Snort 的时候以命令行交互的方式进行选择，现在可供选择的日志形式有三种，报警形式有五种。

Snort 可以把数据包以解码后的文本形式或者 tcpdump 的二进制形式进行记录。解码后的格式便于系统对数据进行分析，而 tcpdump 格式可以保证很快地完成磁盘记录功能，而第三种日志机制就是关闭日志服务。

报警信息可以发往系统日志，也可以用两种格式记录到报警文件中去，或者通过 Samba 发送 WinPopup 信息。发送到报警文件的警告格式分为完全和快速两种格式，完全警告是将报文的报头信息和警告信息全部记录下来，而快速方式只把报头中的部分信息记录下来，以便提高记录效率。系统日志的警告信息可以用 swatch 等工具很方便地进行监视。WinPopup 警告信息则可以很方便地在 Win 95 以上系统的桌面进行显示。同样，第五种方法就是关闭报警。

7.1.2　底层库的安装与配置

安装 Snort 所需的底层库有三个，如下所示：

- Libpcap，Libpcap 提供的接口函数主要实现和封装了与数据包截获有关的过程，它是由

Lawrence Berkeley National Laboratory 创作的。过去只能支持 UNIX,现在已经可以支持 Windows。

Snort 所需的版本大于或等于 0.4.0。

- Libnet,Libnet 提供的接口函数主要实现和封装了数据包的构造和发送过程。Libnet 主要由 Mike D Schiffman 设计和维护。

Snort 所需要的版本为 1.0.2。

- NDIS packet capture Driver,NDIS packet capture Driver 是为了方便用户在 Windows 环境下抓取和处理网络数据包而提供的驱动程序。Packet Driver 分为 9x、NT、2000 三种不同类型。

在 Linux 和 Solaris 环境下,必须首先安装 Libpcap,然后才能安装 Snort,如果需要还应该安装 Libnet;在 Windows 环境下,必须首先安装 NDIS packet capture Driver,然后才能安装 Snort。

1. Libpcap 的安装

在 Linux 和 Solaris 环境下,Libpcap 的安装过程基本一样,具体步骤如下:

(1) 检查 Libpcap

检查系统中 Libpcap 是否存在,因为在某些 Linux 发行版本(如 RedHat 7.0)的完全安装过程中,系统会自动安装 Libpcap。

为确定 Libpcap 是否存在,检查在/usr/local/lib/目录下是否已有 Libpcap.a 文件,在/usr/include/pcap/目录下是否已有 pcap.h、pcap-namedb.h 文件以及 net 子目录,在/usr/include/pcap/net/目录下是否已有 bpf.h 文件。如果上述文件均存在,就说明 Libpcap 安装成功。

(2) 解开压缩包

采用如下命令解开压缩包:

```
# tar-zxvf libpcap.tar.z
```

(3) 正式安装

采用如下命令进行安装:

```
# cd libpcap
#. /configure
# make
# make check
# make install
```

至此 Libpcap 的安装过程完成。

2. Libnet 的安装

在 Linux 和 Solaris 环境下 Libnet 的安装过程也是一致的,具体过程如下:

(1) 解开压缩包

采用如下命令解开压缩包:

```
# tar-zxvf libnet-1.0.2a.tar.gz
```

(2) 正式安装

采用如下命令进行安装：

```
# cd libnet-1.0.2a
# ./configure
# make
# make install
```

至此 Libpcap 的安装过程完成。这时可以检查在/usr/lib/目录下是否已有 libnet.a 文件，在/usr/include/目录下是否已有 libnet.h 文件以及 libnet 子目录，在/usr/include/libnet 目录下是否已有 libnet-headers.h、libnet-structures.h、libnet-functions.h、libnet-macros.h、libnet-asn1.h、libnet-ospf.h 文件，在/usr/bin/目录下是否已有 libnet-config 文件。如果上述文件均已存在，就说明安装成功。

3. NDIS packet capture Driver 的安装

NDIS packet capture Driver 的安装在 Win 9x、Win NT 和 Win 2000 环境下有所不同，下面分别进行介绍。

(1) Win 9x

执行下载的自解压文件 Packet95.exe，并把解压后的文件(包括 Vpacket.inf 和 Vpacket.vxd)放在某一临时目录下。打开 Win 9x 控制面板，打开“网络”，单击“【添加】按钮，选择“协议”后单击【添加】按钮，选择从“磁盘安装”，然后选择解压后的目录，单击【确定】按钮，选择“BPF Packet capture Drive for Win 95/98 v X.XX”(X.XX 代表所安装的版本号)。单击【确定】按钮后，重新启动计算机即可。

(2) Win NT

执行下载的自解压文件 PacketNT.exe，并把解压后的文件(包括 Oemsetup.inf 和 Packet.sys)放在某一临时目录下。打开 Win NT 控制面板，打开“网络”选项，选择“协议”后单击【添加】按钮，选择从“磁盘安装”，然后选择解压后的目录，单击【确定】按钮。选择“Network Packet Drive for Win NT v X.XX”(X.XX 代表所安装的版本号)。单击【确定】按钮后，重新启动计算机即可。

(3) Win 2000

执行下载的自解压文件 Packet2K.exe，并把解压后的文件(包括 packet.inf 和 packet.sys)放在某一临时目录下。打开 Win 2000 控制面板。打开“网络和拨号连接”，然后打开“局域网连接”并单击【属性】按钮。在新窗口中选择“安装”以便开始安装新的网络组件。选择“协议”后单击【添加】按钮。选择从“磁盘安装”，然后选择解压后的目录，单击【确定】按钮。选择“BPF Packet capture Drive for Win 95/98 v X.XX”(X.XX 代表所安装的版本号)。单击【确定】按钮后，重新启动计算机即可。

7.1.3　Snort 的安装

Snort 在不同操作系统环境下的安装过程会有所不同，但是都很简单。下面分别作简要介绍。

1. Linux 环境下的安装(确认 Libpcap 已经安装成功)

安装过程如下：

```
# tar-zxvf  snort-1.8.tar.gz
# cd snort-1.8
# ./configure
# make
# make install
```

至此 Snort 安装完成。检查/usr/local/bin/目录下是否已有 Snort 文件存在,存在则表明安装已经成功。

2. Solaris 环境下的安装(确认 Libpcap 已经安装成功)

在 Solaris 下,同样可以按照 Linux 下的步骤和方法使用源代码包进行安装,另外还提供了 Solaris 特有的 Packet Format 包。安装过程如下:

```
# pkgtrans snor-1.8-sol-2.7-sparc-local /var/spool/pkg
# pkgadd
```

在此选择"Mrsnort"选项进行安装即可。

3. Win 32 环境下的安装

Win 32 下的安装也很简单,过程如下:

解开 snort-1.8-win32-source.zip。用 VC++ 打开位于 snort-1.8-win32 - source \ snort-1.7 \ win32 - Prj 目录下的 snort.dsw 文件。选择"Win 32 Release"编译选项进行编译。在 Release 目录下会生成所需的 Snort.exe 可执行文件。

7.1.4 Snort 的配置

在使用 Snort 之前,需要根据网络环境和安全策略对 Snort 进行配置。主要包括:

- 设置网络变量
- 配置预处理器
- 配置输出插件
- 配置所使用的规则集

在此并不需要自己编写配置文件,只需对 Snort.conf 文件进行修改即可。这个文件包含了大量的默认设置,而且都有很好的注释,用户可以方便地对 Snort 进行配置。

首先,对主要的配置项介绍如下:

- HOME _ NET 变量:要对该变量进行设置,找到 var HOME _ NET any,并把 any 换为实际的网络地址,如 192.168.9.0/24,如果网络有多个网段组成,则可以写成[192.168.9.0/24,10.0.0.0/24]的形式。

其他环境变量可以根据现有的的网络环境进行调整,大多数可以使用默认值。

- 预处理器:是在基于规则的模式匹配之前运行的模块,通常为规则匹配进行一些前期的处理,如 IP 分片重组、TCP 流重组、各种应用层的解码等工作。根据这些模块自身的需求可以对其进行配置,通常采用默认的配置。
- 输出插件:主要在报文匹配某条规则需要输出时,调用相应的输出插件;对它的配置也是插件各自定义的。可根据 snort.conf 的说明进行相应的配置。

7.1.5　其他应用支撑的安装与配置

如果用户需要把结果输出到数据库中，就需要安装数据库软件，并进行相应的配置。对数据库的配置可以参见 snort 目录下的 README.database 文件：创建一个数据库，创建一个具有“INSERT”和“SELECT”权限的用户，建立数据库的结构。

7.2　Snort 的使用

Snort 有三种运行方式：嗅探器、抓包器和网络入侵检测系统，这里主要介绍 Snort 作为网络入侵检测系统的使用方法。

7.2.1　Libpcap 的命令行

Snort 和大多数基于 Libpcap 的应用程序一样，可以使用标准 BPF 类型的过滤器。下面将对过滤器的设置做一个简单介绍，详细的说明可以参见 tcpdump 或 Snort 的帮助页（使用命令“man tcpdump”）。

过滤器的正确设置将决定 Snort 是否只看到所感兴趣的数据包，如果没有设置过滤器，将看到所有网络中的数据包。简单地说，过滤器可以限制主机、网络和协议的范围，可以使用逻辑运算符把若干个过滤器联合起来。整个过滤器的表达式由一个或多个元语组成，而一个元语则是由一个或多个关键字加上一个相关的值（字符串或数字）所组成。

关键字分为以下几类：

- 属性类类键字：说明后面所跟值的意义，这样的关键字有 host、net、port。例如：host www.yahoo.com、net 128.5、port 21 等。如果一个元语没有属性关键字，默认为 host。
- 方向类关键字：说明报文的流向，这样的关键字有：src、dst、src or dst、src and dst。例如：src www.yahoo.com、dst net 128.5、src or dst port 21 等，默认为 src or dst。对于空连接层（如：使用 slip 等的点到点协议），可以使用 inbound 和 outbound 来说明方向。
- 协议类关键字：用来限制协议，这样的关键字有：ether、fddi、ip、arp、rarp、decnet、lat、sca、moprc、mopdl、tcp、udp 等。如果一个元语没有协议类关键字，那么所有可能的协议都符合。

另外还存在其他一些特殊的关键字，包括 gateway、broadcast、less、greater 以及算术运算符。

过滤器的描述功能非常强大，可以用来描述任何想要得到的报文。下面是两个过滤器描述报文的例子：

- net 192.168.9 and not host 192.168.9.1：说明除了主机 192.168.9.1 外其他所有 192.168.9 网段的网络数据报文。
- 'tcp[13]& 3 ！ =0 and not src and dst net localnet'：说明涉及到外网的 tcp 会话中起始和终止报文（SYN 和 FIN 报文）。

通过使用过滤器表达式，可以限制 Snort 监控的网络流量，使 Snort 更高效地工作。如果网络中的数据流量很大，Snort 的丢包率可能会很高，这时可以通过组合使用过滤器，在一台机

器上检测一部分网络流量，而在另一台机器上检测其他部分的网络流量。

7.2.2 Snort 的命令行

Snort 的命令行参数很多，可以使用 Snort -？ 命令列出这些参数及其简单的解释，详细的解释可以使用 man snort 命令查看帮助页，或者直接阅读 Readme 文件和 Usage 文件。

下面先对 Snort 的命令行参数进行简单的描述：

- -A <alert>：设置 Snort 的警告模式，alert 可能的取值包括 full、fast 或 none。full 模式进行正常的“典型 Snort”类型的警告；fast 模式只把时间、消息、IP 地址和端口记录到警告文件中；none 模式则关闭报警。
- -a：显示 ARP 报文。
- -b：以 tcpdump 格式将报文记录到日志文件中，报文以二进制的形式记录，因为不需要进行到文本的转换，所以速度更快。
- -c <cf>：使用配置文件<cf>。
- -C：只用 ASCII 显示报文负载，不显示十六进制输出。
- -d：显示应用层数据。
- -D：以守护进程形式(Daemon)运行，除非另行说明，警告将被发送到/var/log/snort/alert 中去。
- -e：显示第二层的包头数据(默认不显示)。
- -F：<bpf>：从文件<bpf>中读取 BPF 过滤器。
- -g <gname>：Snort 初始化后用户组标志转换为<gname>，这种转换使 Snort 抛弃初始化时必需的超级用户权限，是一种更安全的措施。
- -G：为了与 1.7 兼容而使用的选项。
- -h <hn>：设置“home network”到<hn>。
- -i <if>：在网络接口<if>上监听。
- -I：添加网络接口名称到警告输出。
- -k <checksum mode>：设置报文头检验和检查，可选的值为 all、noip、notcp、noudp、noicmp、none。
- -l <ld>：将日志文件放到目录<ld>中。
- -L <fn>：设置二进制输出的文件名为<fn>。
- -m <mask>：设置所有的 Snort 输出文件的访问掩码(Umask)为<mask>。
- -M <wkstn>：发送 Windows 消息到包含在<wkstn>文件中的工作站列表，这个选项需要安装 samba 服务。
- -n <num>：在处理完<num>个报文后退出。
- -N：关闭日志功能，警告功能仍然正常。
- -o：改变标准规则应用到报文上的顺序，从标准的 Alert -> Log 到 Pass -> Alert -> Log，可以避免为了过滤 alert 规则而使用巨大的 BPF 参数。
- -O：使用 ASCII 码输出时，使用“xxx.xxx.xxx.xxx”代替 IP 地址，如果使用 -h 定义了 homenet，则本地网的 IP 地址进行这种替换，而外部地址则不替换。主要是为了方便

在交流日志信息的时候可以隐藏具体的细节。

- -p:关闭混杂(Promiscuous)监听模式。
- -P <snaplen>:设置 Snort 的抓包截断长度。
- -q:在安静模式运行,不显示标题和状态统计。
- -r <tf>:从 tcpdump 格式的报文文件中读取数据,而不是从网络上监听。
- -s:使用 syslog 日志警告信息。
- -S <n=v>:使用这个选项定义变量 n=v,和在规则文件中定义的效果相同,如果命令行和规则文件中对同一变量进行定义,则使用命令行的定义。
- -t <chroot>:初始化后改变 Snort 的根目录到目录<chroot>,需要注意的是所有的 log/alert 文件名都是相对于 chroot 目录的。
- -T:Snort 进入自检模式,检查命令行和传入的规则文件的正确性。
- -u <uname>:初始化后改变 Snort 的用户 ID 到<uname>。
- -U:打开 UTC 时间戳。
- -v:更详细地输出。
- -V:显示版本号,然后退出。
- -x:遇到较小的 IPX 数据包时显示消息。
- -X:显示链路层上的原始数据。
- -y:在报文时间戳上显示年份。
- -z:设置 Snort 警告的确信模式。
- -?:显示 Snort 的简要使用说明,然后退出。

一个常用的命令组合如下所示:

```
./snort -dev -l./log -h 192.168.9.0/24 -c  snort.conf
```

通常,不需要屏幕显示,也不必记录数据链路层数据,因此可以省略-d 和-v 参数:

```
./snort -d -h 192.168.9.0/24 -l./log -c snort.conf。
```

7.2.3　高性能的配置方式

如果在一个高数据流量(比如大于 100Mb/s)的网络环境下运行 Snort,就需要考虑如何配置 Snort 才能使它高效率地运行,这就要求使用更快的输出功能,产生更少的警告,可以使用诸如-b、-A fast、-s 等选项。

例如:

```
./snort -b -A fast -c snort-lib
```

使用这种配置选项,日志信息将被以二进制的 tcpdump 格式记录到 Snort.log 文件中,然后,使用下面带有-r 选项的命令读取这个文件,做进一步的分析:

```
./snort -d -c snort-lib -l./log -h 192.168.9.0/24 -r snort.log。
```

7.3 Snort 的规则

如果只有 Snort 的可执行程序，而没有规则文件，Snort 就不能完成真正的入侵检测功能。规则文件就是 Snort 的攻击知识库，没有规则 Snort 就不能识别任何攻击。每条规则包含一种攻击的攻击标识，Snort 就是通过它来识别攻击。这些规则被分类放在 Snort 源码所在目录 *.rules 文件中，如 web.rules、ftp.rules 等。

7.3.1 规则的语法

Snort 是一种基于网络的入侵检测系统，主要用于检测来自网络的攻击。Snort 规则就是使用“一种简单的、轻量级的描述语言”来描述网络上带有攻击标识的数据包。这种描述语言在形式上不够完备，但作为一种简单的、轻量级的语言已经相当灵活，而且具有很强的描述能力。

Snort 的规则在逻辑上分为两部分：规则头(Rule Header)和规则选项(Rule Option)。规则头定义了规则的行为(Action)、所匹配网络报文的协议、源地址、目标地址及其网络掩码、源端口和目标端口等信息；规则选项部分则包含了所要显示给用户查看的警告信息以及用来判定此报文是否为攻击报文的其他信息(比如：tcp 的 flag 字段，数据字段的内容等)。

下面是一个简单的规则：

```
alert tcp any any -> 192.168.9.0/24 111(content: "|00 01 86 a5|"; msg: "mountd access";)
```

在这个例子中，括号左面为规则头，括号中间的部分为规则选项，规则选项中冒号“:”前的部分称为选项关键字(Option Keyword)，这里需要注意的是规则选项部分并不是任何规则都必需的，它只是用来明确定义出表示某种攻击、需要采取某种行为(如警告等)的报文。只有当组成规则的各个元素都为真时才能触发相应的操作，综合考虑，限制报文的各元素是逻辑与的关系；同时，规则库中的各条规则可以被考虑为一个大的逻辑或的关系。在详细讨论规则的各部分之前，先看一下规则文件的语法。

1. 规则文件的语法

规则分类存放在规则文件中。规则文件是普通的文本文件，可以使用普通的文本编辑器打开进行编辑。规则文件的命名没有明确规定，当前默认的规则是“类名 .rules”，snort.conf 也是规则文件。规则文件可以包含注释行，注释行以“#”引导注释，但可以使用注释语句注释此规则文件或某条规则。如果某条规则不适合你的网络环境，如经常报告错误的警告信息，就可以使用“#”注释掉此规则。

为了方便使用，Snort 允许定义变量，并在规则中使用这些变量。通过变量的使用，可以避免繁琐地重复输入相同的字符串。如果使用变量来表示某些可能会改变的量，则可以简化规则的修改过程，只要修改变量的定义即可，而不用逐行进行修改。变量的定义格式如下所示：

```
var: <name> <value>.
```

在规则中可以直接使用 $ <name>，而遇到变量名解释器就会使用<value>替代 $ <name>。下面是一个典型的例子：

```
var MY __ NET[192.168.9.0/24,10.1.1.0/24]
```

这是一个比较典型的使用方式，$HOME __ NET 和 $EXTERNAL __ NET 是经常使用的两个环境变量，分别用来定义内部网络地址和外部网路地址。在规则文件中，如果重复定义了一个变量，后面定义的值就会取代以前的定义。此外，还存在一些方式可以对变量名的定义进行修改，可以使用“$”定义元变量，也可以在定义中使用修改操作符“?”和“-”。下面举例对它们进行解释：

- $name：定义一个元变量。
- $(name)：使用变量 name 的内容替代。
- $(name:-default value)：使用 name 的内容替代。如果 name 未定义，则使用 default value 替代。
- $(name:? message)：使用 name 的内容替代，如果 name 未定义，则打印错误消息 message，并退出程序。

下面是一个具体的例子：

```
var MY __ NET $(MY __ NET:-192.168.9.0/24)
log tcp any any -> $(MY __ NET:? MY __ NET is undefined!)23
```

第一句定义了一个变量 MY __ NET，如果已经定义了 MY __ NET，则保持原值不变。如果未定义，就为 192.168.9.0/24。规则定义中对变量引用的含义是，如果 MY __ NET 已经定义，则使用其值进行替换，否则报错“MY __ NET is undefined”并退出程序。

另一个可以在规则文件中使用的关键字是 include，它允许规则文件引入其他的规则文件，并放到当前位置。解释器遇到此关键字时就会先去解释它所包含的文件，就像规则文件的当前位置包含了其他文件一样，它的工作方式和 C 语言的“#include”相似。语法如下所示：

```
include:<include file path/name>
```

综合使用这两个机制可以灵活方便地书写规则文件。下面将详细介绍规则各部分的语法。

2. 规则头

规则头包含一个报文关键的地址信息、协议信息以及当报文符合此规则时各元素应该采取的行动。

(1) 规则行为

规则头的第一个字段就是规则行为(Rule Action)。Snort 中定义了五种可选的行为：alert、log、pass、activate、dynamic。其语义如下：

- alert：使用设定的警告方法生成警告信息，并记录这个报文。
- log：使用设定的记录方法记录这个报文。
- pass：忽略这个报文，这个方法初看没有什么用处，因为一个报文如果不匹配，所有的规则就会自动被忽略。这种规则主要应用的场景为：下面一条或几条规则限定了某类报文，但其中又夹有几种正常的报文，这样就可以在这些规则之前使用 pass 行为来定义这几种报文的规则。
- activate：进行 alert，然后激活另一个 dynamic 规则。

● dynamic:等待被一个 activate 规则激活,然后进行 log。

此外,还可以定义自己的规则类型,并和一个或多个输出插件相关联,然后在规则文件中把规则类型当作规则行为字段进行使用。每个行为都应该和设定的输出插件相关联,如果想对某些规则使用特定的输出插件,而不是默认的输出行为,就可以定义自己的规则类型。例如默认的 log 行为是记录到 syslog 中,但对于某些规则如果需要以 tcpdump 格式记录,那么就可以采用如下的方式来定义这样的规则类型:

```
ruletype suspicious
{
        type log
        output log _ tcpdump:suspicious.log
}
```

下面的例子创建了一个既记录到 syslog 又记录到 mysql 的规则类型。

```
ruletype redalert
{
        type alert
        output alert _ syslog:LOG _ AUTH LOG _ ALERT
        output database:log,mysql,user=snort dbname=snort host=localhost
}
```

(2) 协议字段

下一个域是协议字段(Protocol)。当前 Snort 支持三种 IP 协议——TCP、UDP 和 ICMP,将来可能会支持更多的协议,如 ARP、IGRP、GRE、OSPF、RIP、IPX 等。这个字段中可能的值包括 tcp、udp 和 icmp。

(3) 地址和端口信息

规则头的下一部分是描述规则的 IP 地址和端口信息。关键字 any 用来定义任意 IP 地址。由于 Snort 不提供主机名的查询机制,所以必须使用 IP 地址加上一个 CIDR(Classless Inter Domain Routing)块来表示地址,CIDR 块用来说明 IP 地址的网络掩码。/24 说明一个 C 类地址,/16 说明一个 B 类地址,/32 指定一个主机。如 IP 地址和 CIDR 块的结合 192.168.9.0/24 就表示从 192.168.9.1 到 192.168.9.255 的 IP 地址范围。

操作符"!"可以应用到 IP 地址上,表示除了此 IP 地址范围外所有的地址。

IP 地址字段也可以是 IP 地址列表,IP 地址列表由使用逗号分隔的 IP 地址和 CIDR 块组成,并放在方括号中。例如:

```
Alert tcp ! [192.168.9.0/24,10.1.1.0/24]any -> [192.168.9.0/24,10.1.1.0/24]111(content:
"|00 01 86 a5|";msg:"external mountd access";)
```

端口号部分可以使用很多方法来说明:

- 关键字 any 说明任意端口。
- 一个数字指定静态的端口。
- 使用冒号隔开的两个数字表示端口的范围,如:1:1024 表示从 1 到 1024 的端口,1:6000

表示从 1 到 6000 的端口。

- 同样可以使用非操作符“!”。

还可以使用方向操作符来限制数据报的流向。“->”表示从左端流向右端的数据报文，“<-”则反之，而“<>”用来匹配双向的数据流。下面的规则可以用来记录以某个范围内的主机(192.168.9.0/24)作为服务器(telnet 服务的端口是 23)的所有 telnet 会话的双向数据流。

```
log! 192.168.9.0/24 any <> 192.168.9.0/24 23
```

这种双向的记录一般用来对某个协议的全过程进行记录或分析，如 POP3 协议，但是它所产生的数据量会较大，所以应该小心使用。

这里要强调两种规则：activate 和 dynamic 规则。这两种规则的组合赋予 Snort 很强大的能力：可以使用一条规则激活另一条规则。它适用于这种情况：当某种攻击发生后需要记录两个或多个包时，它可以将检测的状态“保持”而跨越多个包。activate 规则和 alert 规则相似，只是增加了必须存在的域“activate”，而 dynamic 则必须有一个域 activated _ by，此外还要有另外一个域“count”。当一个 activate 规则被触发后，就会激活相关的 dynamic 规则(它们的 activates 和 activated _ by 域的值相同)对后续的“count”个包进行记录，看下面的例子：

```
activate tcp ! $HOME _ NET any -> $HOME _ NET 143(flags:PA; \
count:"|E8C0FFFFFF| \bin|;activates:1; \
msg:"IMAP buffer overflow!";)
dynamic tcp ! $HOME _ NET any -> $HOME _ NET 143(activated _ by:1;count:50;)
```

这个例子表示在检测到 IMAP 缓冲溢出攻击时进行告警，然后记录后续的外网主机向本地网主机 143 端口发送的 50 个报文。这种做法基于下面的考虑：如果缓冲区溢出攻击成功，那么接下来传输的 50 个报文中很有可能包含比较有用的数据，所以记录这些报文有利于事后的分析。

3. 规则选项

对规则选项的分析构成了 Snort 检测引擎的核心，既易于使用又能使功能更加强大灵活。所有选项使用分号“;”分隔，选项的关键字和它的值之间使用冒号分隔。目前有三十几种选项关键字，简述如下：

- mag：打印一条警告信息到警告或日志中。
- logto：记录符合这条规则的所有报文到到用户指定的文件中而不是标准的输出文件。注意当 Snort 处于二进制记录模式时，这个选项无效。
- ttl：检查 IP 报文的 TTL 域的值。
- tos：检查 IP 报文的 IOS 域的值。
- id：检查 IP 报文的分片 ID 域的值。
- ipoption：检查 IP 报文的 option 域的值。
- fragbits：检查 IP 报文的分片比特位的值。
- dsize：检查 IP 报文的负载长度的值。其格式为：dsize:[>|<] <number>。
- flags：检查 TCP 的 flags 域。
- seq：检查 ICP 的序列号域的值。

- ack:检查 ICP 的确认域的值。
- itype:检查 ICMP 协议的类型域的值。
- icode:检查 ICMP 协议的代码域的值。
- icmp __ id:检查 ICMP 回应消息的 ID 域的值。
- icmp __ seq:检查 ICMP 回应消息的序列号域的值。
- content:在报文负载中搜索某个模式。这个字段是 Snort 的一个重要特性。进行比较的数据可以包含二进制数据,二进制数据使用"|"括起来的十六进制字节串表示,如:content:"|90C8 C0FF FFFF|/bin/sh";。除非指定 nocase,否则这种匹配是大小写敏感的。另外需要注意的是这种比较非常耗费资源,一定要增加尽可能多的其他选项来限制进行搜索的报文。
- content - list:在报文负载中搜索一个模式的集合。
- offset:调整 content 选项,设置模式匹配开始的偏移量。
- depth:调整 content 选项,设置模式匹配的最大搜索长度。
- nocase:进行内容匹配时,字符串的大小写不敏感。
- session:对一个指定的会话记录其应用层数据。
- rpc:审查 RPC 服务对某个应用或过程的调用。这个选项审查解码后的 RPC 请求,它的格式是:rpc:<number>,[number|*],[number|*]>。
- resp:激活响应(如:关闭连接等)。这个选项实现便利的响应,它的格式是关闭连接方式的列表,如 resp:<resp __ modifier[,resp __ modifier...]>;,可选的方式包括 rst __ snd(向发送方发送 TCP __ RST 报文)、rst __ rcv(向接收方发送 rst 报文)、rst __ all(向通信双方都发送 rst 报文)、icmp __ net(向发送方发送网络不可达的 icmp 报文)、icmp __ host(发送主机不可达的 icmp 报文)、icmp __ port(发送端口不可达的报文)、icmp __ all(发送所有上述的 icmp 报文)。
- react:激活响应(如:阻塞网站的访问等),这是一种更高级的反应机制。当前可获得的方式包括 block(关闭连接并发送一条提示信息)、warn(发送一条警告信息),附加的调整包括 msg(在消息中显示 msg 选项中的内容)、proxy(通过代理发送消息)。

在 1.8 版本中新增了如下的关键字。

- sid:用来惟一标识 Snort 的规则。当前 sid 的范围是这样确定的:<100,Snort 保留;100~1000000,Snort 自身规则使用;而用户使用>1000000 的 sid 标识自己定义的规则。
- rev:表示版本修订的次数,总是和 sid 联合使用。
- classtype:指定这个攻击的类型,不同的类型具有不同的级别。
- prioroty:指定此规则的严重级别,每个攻击类型都有一个默认级别。
- unicontent:允许在一个请求的 URI 进行搜索匹配特定的格式。它和 content 相似,不过只是在 URI 中搜索。
- tag:允许规则不只是记录触发此条规则的报文,还可以记录其后的网络数据,以方便事后分析。它的格式是:tag:<type>,<count>,<metric>,[direction]。Type - session(记录所有本次会话的内容),host(记录所有来自攻击机器的网络数据);记录 count 个 metric(packet:报文或 seconds:秒);direction 只有 tag 为 host 时才有效(src 表示来自该

主机的，dst 表示到达该主机的）。

- ip__proto：测试 IP 头的协议字段，内容是/etc/protocol 中的协议名称。
- sameip：用来检查源地址是否和目标地址相同。
- stateless：和 stream4 预处理器联合使用，允许规则的匹配不考虑连接的状态，格式为 stateless。
- regex：指明在 content 中能够包含通配符，但注意不是真正的正规表达式，只是通配符（* 表示任意长字符串，? 表示一个字符）而已。

这里的规则选项其实对应着源代码中的各个处理插件。

4. 预处理器

传统的基于规则的检测有时候不能满足某个特殊的需求，通过预处理器就可以用一种特殊的方式对报文进行分析和修改。预处理器会在检测引擎之前、报文解码之后运行。预处理器在 1.5 版本之后才引入。

对预处理器的调用和配置使用 preprocessor 关键字，它的格式是：preprocessor <name>：<options>。

下面简单介绍 Snort 所提供的预处理器。

- Minfrag：用来检查分片的报文是否小于某个值，通常路由器造成的分片的长度不小于 512 个字节，如果太小就说明有人想通过分片掩藏网络传输的数据。它的格式是：am-infrag：< threshold number >。Snort1.8 版本中已经不再使用 Minfrag，它的功能由 Stream4 实现。
- HTTP Decode：通常在 HTTP URI 中允许包含一些编码过的字符，攻击者可以通过这种方式躲避那些基于内容审查的检测方式。这个预处理器处理 HTTP URI 字符串，把它转化为没有歧义的 ASCII 码字符串。它的格式是：http__decode：<port list> [-unicode] [-cginull]。
- Portscan Detector：端口扫描的意图众所周知，所以应该像防止攻击一样防止端口扫描。端口扫描检测预处理器主要完成两件工作，一是记录来自某个 IP 地址的端口的开始和结束，二是在指定了日志文件的情况下，将扫描类型、目的地址和端口等全部记录下来。

在 Snort 中，端口扫描的一种描述是在 T 秒时间内对超过 P 个端口的 TCP 连接企图或者是在 T 秒时间内向超过 P 个端口发送 UCP 数据报。这些目标端口可以是同一个目标 IP 地址上的不同的端口，也可以是不同 IP 地址的相同端口。当前版本的预处理只能处理单对单和单对多两种形式的端口扫描，后期的版本将会增加分布式端口扫描的检测（例如，多对单和多对多方式的扫描）。端口扫描又可以描述成一个隐秘扫描（Stealth Scan）的数据报，包括 NULL、FIN、SYNFIN、XMAS 等，通过对这些特定数据报的解释同样可以发现端口扫描。它的格式是：portscan：< monitor network > < number of ports > < detection period > <file path>。

- Portscan Ignorehosts：<host list>：这个预处理器是用来对端口扫描检测预处理进行修正，忽略某些机器发出的 TCP、SYN 和 UDP 扫描，这些扫描报文对于 NTP、NFS 和 DNS 服务器来说可能是正常的。它的格式是：portscan-ignorehosts：<host list>。
- Defrag：这个预处理器完成 IP 分片重组的功能，这样就可以发现那些想要通过简单的分

片试图绕过检测的攻击行为。Defrag 实现了 minfrag 的功能，使用它就不需要再使用 minfrag；但它自己也有了升级版本 Frag2。它的格式是：defrag，无需参数。

- Frag2：是 1.8 版本中新引入的预处理器，用来替换 Defrag。它比 Defrag 能够更高效地使用内存，而且使用与其他 Snort 组件相同的内存管理程序。Frag2 需要配置最大的内存限制和超时值（如果不指定，则它们的默认值分别是 4M 和 60 秒）；它的格式是：frag2：[memcap <xxx>]，[timeout <xx>]。
- Stream2：该处理器主要完成 TCP 流重组的功能。它的格式是：stream2：timeout <timeout>，ports <ports>，maxbytes <maxbytes>。
- Stream4：这个预处理器提供了 TCP 流重组和状态分析的功能。当前 Stream4 可以同时处理 256 个 TCP 数据流，以后将可以同时处理 64000 个 TCP 数据流。Stream4 又有两种可以配置的使用模式：一种是 Stream4 预处理模式，另一种是 Stream4 流重组插件模式。Stream4 预处理器模式的格式为：preprocessor stream4：[noinspect]，[keepstats]，[timeout <seconds>]，[memcap <bytes>]，[noalerts]。

其中，noinspect 是取消状态审查；keepstats 是记录会话信息到<logdir>/session.log 文件中；timeout <seconds>是在列表中保留一个活动的数据流的时间，默认为 30 秒；memcap <bytes>是最大消耗的内存数，默认是 8MB；noalerts 是关闭对流事件的告警。

流重组插件模式的格式为：stream4 _ reassemble：[clientonly]，[serveronly]，[noalerts]，[ports <portlist>]。

其中，clientonly 是只重组客户端的数据；serveronly 是只重组服务器端的数据；noalerts 是关闭报警；ports <portlist>是进行重组的端口列表（all 表示所有端口，默认为 21、23、25、53、80、110、111、143 和 513）。

- Spade：Spade 用来进行基于统计的异常检测，基本上是一个全新的检测引擎（与其他的检测引擎完全不同）。它是由第三方所开发的。

5. 输出模块

Snort 的输出模块实现了对检测结果的格式化表示和输出。它处于预处理和检测完成之后，在系统调用的日志和告警子系统中运行。它在规则文件中的表示形式与预处理器十分相似。在规则文件中相同的类型可以定义多个输出插件，如果有事件发生，这些输出插件模块就会按照顺序运行。在默认情况下，输出模块会把数据发送到/var/log/snort 目录或者用户指定（使用 -l 参数指定）的目录下。

输出模块在运行的时候会被动态加载，在规则文件中使用下面的语法来说明需要加载的模块：output <name>：<options>。

下面简单介绍 Snort 所提供的输出模块。

- Alert _ syslog：这个模块把警告发送到 syslog 系统日志中，这和通过命令行输入 -s 命令选项所实现的功能相似，但它还允许用户在 Snort 规则文件中对日志属性和优先级进行设置，给用户很大的灵活性。

当前设置关键字包括三类：第一类为选项，包括 LOG _ CONS、LOG _ NDELAY、LOG _ PERROR、LOG _ PID。第二类为属性，包括 LOG _ AUTH、LOG _ AUTHPRIV、LOG _ DAEMON、LOG _ LOCAL0、LOG _ LOCAL1、LOG _ LOCAL2、LOG _ LOCAL3、LOG _

LOCAL4、LOG _ LOCAL5、LOG _ LOCAL6、LOG _ LOCAL7 和 LOGUSER。第三类为优先级，包括 LOG _ EMERG、LOG _ ALERT、LOG _ CRIT、LOG _ ERR、LOG _ WARNING、LOG _ NOTICE、LOG _ INFO、和 LOG _ DEBUG。

它的格式是：alert _ syslog：<facility> <priority> <options>。

- Alert _ fast：这个模块的 fast 是相对于 Alert _ full 模块而言的，因为它不需要记录所有的报文头信息，只是迅速地显示一行警告信息到输出文件中，输出文件名也就是惟一需要配置的项。它的格式为：alert _ fast：<output　filename>。
- Alert _ full：这个模块不仅显示警告信息，同时还显示整个报文头。因为它需要把大量的数据解释为格式化的数据进行输出，所以相对较慢。它的格式是：alert _ full：<output filename>。
- Alert _ smb：这个模块向指定的 NETBIOS 命名主机发送 WinPopup 警告信息。如果有警告信息，这些主机就会弹出一个窗口来显示。主机名被保存到一个文件中，所以在配置时需要指明这个文件名。它的格式是：alert _ smb：<alert workstation fileneme>。
- Alert _ unixsock：创建一个 UNIX 域的套接字，并向它发送警告数据。外部的应用就可以从这个套接字中实时地接收到警告数据。它的格式是：alert _ unixsock。
- Log _ tcpdump：这个模块以 tcpdump 格式文件的形式记录数据报文。一方面因为 tcpdump 格式记录的速度较快，另一方面因为存在很多分析 tcpdump 格式文件的工具，所以先用这个模块来记录报文数据，然后再进行事后的分析。这个模块中惟一需要指明的参数就是 tcpdump 格式文件的名称。它的格式是：log _ tcpdump：<output filename>。
- XML：这个模块用 SNML 格式来记录日志信息，可以记录到本地文件或者通过网络的远程记录中心。通过使用这个模块，Snort 可以向中央数据库发送日志信息，这样可以在网络中构建一个非常灵活的入侵检测框架；使用这个模块，甚至还可以自动地将报告发给 CERT 中心，或者发给企业内部的反应小组。它的格式为：Output xml：[log | alert]，[parameter list]。
- Database：数据库模块可以把 Snort 的数据发送到很多 SQL 数据库中，目前支持四种类型的数据库：MySQL、PostgreSQL、Orcle 和 UNIX ODBC 兼容的数据库。它的格式是：database：<log | alert>，<database type>，<parameter list>。
- CSV：个模块把警告数据记录成一种易于导入到数据库的格式。它的格式是：output CSV：<filename> <format>。前者指定输出的文件名称，后者指定输出的格式。
- Unified：这个模块的目的是为了尽可能快地进行 Snort 事件的输出。它以二进制的形式把事件信息输出到两个文件中，其中警告文件中包含事件的高层细节（地址、协议、端口等），日志文件则包含详细的报文信息。它的格式有两种：output alert _ unified：<file name>和 output log _ unified：<file name>。

7.3.2　常用攻击手段对应规则举例

本节介绍一些常用攻击手段所对应的规则，看看如何使用这些规则语法去描述一个攻击。

(1) 最初微软发布 IIS 时,附带的示例网页中存在漏洞,而使用 IIS 的网站管理员经常会忘记删除微软的示例网页(这些网页在 web 根目录的/site/iisamples 目录下),那么如何检测出利用这种漏洞进行的攻击或者攻击的企图呢? 注意,这种请求的 URL 中一定包含"/site/iisamples"字符串,因此可以通过在 URL 中查找/site/iisamples 来检测这种攻击。相应的规则已经在 web - iis.rules 文件中有所描述,如下所示:alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg:"WEB - IIS site/iisamples access";flags:A+;uricontent:"/site/iisamples";nocase;classtype:attempted-recon;sid:1046;rev:1;)。

(2) 红色代码蠕虫(Code Red)。现在来分析一下它的一个变种 Code Red II。它和 Code Red 蠕虫一样,也是利用了微软 IIS web 服务器的一个远程漏洞"微软 Index Server(.ida/idq) ISAPI 扩展远程溢出漏洞(MS01-033)"。微软 IIS 中带有一个索引服务器(Index Server,在 Windows 2000 下名为"Index Server"),在默认安装时,IIS 支持两种映射脚本:管理脚本(.ida 文件)和 Internet 数据查询脚本(.idq 文件),这两种脚本都由一个 ISAPI 扩展 idq.dll 来进行处理和解释。由于 idq.dll 在处理某些 URL 请求时存在一个未经检查的缓冲区,如果攻击者提供一个特殊格式的 URL,就可能引发缓冲区溢出。因此,通过精心构造地发送数据,攻击者可以改变程序执行的流程去执行任意代码。成功地利用这个漏洞,攻击者就可以远程获取本地系统的权限。

下面所示的规则就能够有效地检测到这种攻击,它是在 uri 中查找对 .ida? 的访问来进行攻击的检测。具体规则如下所示:alert tcp $EXTERNAL any -> $INTERNAL 80 (msg:"IDS552/web-iis_IIS ISAPI Overflow ida";dsize: >239;flags:A+;uricontent:".ida?";classtype:system-or-info-attempt;reference:arachnids,552;)。

(3) 下面介绍一个对 Red Hat Linux 进行远程攻击的例子。它是利用 Red Hat 7.0 的 LPRng 包(一个伯克利的 lpr 脱机打印攻击的实现)中的一个函数调用缓冲区溢出的漏洞来进行攻击。在网络流量中,可以通过对导致缓冲溢出的数据进行扫描来实现检测。这条规则在 exploit.rules 中有所描述,如下所示:alert tcp $EXTERNAL_NET any -> $HOME_NET 515 (msg:"EXPLOIT LPRng overflow";flags:A+;content:"|43 07 89 5B 08 8D 4B 08 89 43 0C B0 0B CD 80 31 C0 FE C0 CD 80 E8 94 FF FF FF 2F 62 69 6E 2F 73 68 0A|";reference:bugtraq,1712;classtype:attempted-admin;sid:301;rev:1;)。

现在,我们可以看出 Snort 的优点:通过简单地增加新的规则,Snort 就可以检测到新的攻击。

对规则的更新有以下三种方式:

- 首先,要经常访问 Snort 的官方网站,更新它所发布的新规则。这些规则通常有一定的通用性和稳定性,但时效上可能要弱一点。
- 其次,可以加入 Snort 的邮件列表,它会更及时地根据当前流行的安全漏洞,发布相应的攻击标识以及相应的 Snort 规则。
- 最后,可以根据自己的环境定制自己的规则,或者根据自己发现的新攻击来编写相应的规则。前面两种更新方式能发现很多攻击的类型,但可能只是企图,对用户自己的系统并没有真正的危害;而有些真正威胁到用户系统的攻击,却没有相应的 Snort 规则;还有一些违反安全政策的行为,并不是某种攻击或这种攻击还未发现,这就需要用户自己

去编写自己的规则。这种方式最符合用户的需求，也最有效。

7.3.3　规则的设计

正如上一节最后所介绍的，自己定义的规则更能符合自己环境的需求。本节就首先介绍如何根据自己的环境定制相应的规则，保证不违反网络的安全策略；然后简要叙述如何发现新的攻击，并编写它的攻击标识和相应的 Snort 规则。

根据网络的安全政策设计自己的规则：

(1) 针对具体的网络环境制定自己的安全政策，也就是完全摸清网络的环境、网络提供的服务以及这些服务的允许和禁止要求，然后才能够方便地制定自己的规则。可以把所有增加的规则放到一个单独的规则文件(如：myrule.rules)中，然后把它包含到 snort.conf 这个总体的规则文件中(例如，在最后一行加入 include myrule.ules)。它们的 sid 应设为大于 1,000,000 的数字。

(2) 考察网络中所不提供的服务和提供的服务以及谁可以访问这些服务。

如果有一个内部网络 192.168.9.0/24 不允许从外部网络访问，就可以增加一条规则发现这种异常的访问：alert tcp $EXTERNAL_NET any -> 192.168.9.0/24 any(msg:"Policy:external net attempt to access 192.168.9.0/24";classtype:attempted-recon;sid:10002;rev:1;)。

如存在一台只供内部使用的 WWW 服务器 192.168.8.2，那么所有外部网络对这个服务的访问都是不正常的，可以再增加一条规则：alert tcp $EXTERNAL_NET any -> 192.168.8.2/32 80(msg:"Policy:external net attempt to access 192.168.8.2:80";classtype:attempted-recon;sid:10001;rev:1;)。

某台外部邮件服务器 192.168.7.2 只用来提供邮件服务，另外开了个 ssh 的端口让内部网络管理员进行访问。所有对其他端口的访问都应该是不允许的，外部对 ssh 的访问也是异常的，所以可以加入下面的规则：alert tcp any any -> 192.168.7.2/32! [25,110,22](msg:"Policy:Attempt to access 192.168.7.2";classtype:attempted-recon;sid:10003;rev:1;)。
alert tcp $EXTERNAL_NET any -> 192.168.7.2/32 22(msg:"Policy:External attempt to access 192.168.7.2 ssh";classtype:attempted-recon;sid:10004;rev:1;)。

对其他服务也可以进行相似的控制。例如，不能让远端用户通过浏览器浏览到 HTTP 服务器上真正目录下的文件，就可以在 HTTP 的根目录下建立一个有特别名字的文件(如：FileNameSecrete)，然后在通信流中扫描这个名字，规则如下：alert tcp $EXTERNAL_NET any -> $HTTP_SERVER 80 (msg:"Policy:External attempt to explore directory";uricontent:"FileNameSecrete";classtype:attempted-recon;sid:10005;rev:1;)。

通过上述方法可以方便地检测出对网络安全政策的违反行为，这样就可以迅速得知所有异常的网络访问。

但是要发现一个新攻击，就没有这么容易。这里只是简单介绍如何由一个攻击描述推导出一个攻击标识的过程。

(1) 首先获得某个攻击的描述，这个描述可以来自某些安全列表(如：bugtraq)，有些列表对安全漏洞有着非常详细的描述，甚至还有攻击代码；也可以来自某些黑客网站或是某些黑客

的攻击程序;甚至还有用户本身就是一个受害者,在发现被攻击后,通过检查自己记录的网络数据(使用 tcpdump 等工具)来得到攻击的描述。

(2) 然后可以分析这个描述与正常网络通信的区别。正常网络通信数据应该选取相同的场景,如同样的服务程序、相似的网络访问(通过 Internet 或 Intranet 或本地)、相似的应用场景(相似的应用层会话)。要是有攻击的代码,就可以简单地分析它的源码。查看设置了哪些项、传输了哪些数据、找出与正常情况的差别、缺少这些差异攻击就不能实现等,这些差别可能很多,但想要检测出这种攻击的所有变种就要选取关键的差异。这一步难度较高,如果规则设计得不好,漏报和误报的可能性就会非常大。因此,最好能够与其他管理员进行交流。

(3) 最后使用 Snort 的规则语法来描述这些差异,这个工作相对于上两步应该简单一些。

7.4 Snort 总体结构分析

Snort 是一个非常优秀的、开放源代码的入侵检测系统。从检测机制上看,它不仅具有基于规则的误用检测方法,而且还有基于异常的检测方法(可由第三方添加);从体系结构上看,它充分考虑了扩展的需求,大量使用了插件机制;从功能模块上看,各个模块功能明晰,相对独立,设计合理;从编码上看,它具有很好的编码风格和详细的注释,易于理解。因此,无论从入侵检测实现技术还是编程技术来看,Snort 都是一个非常好的学习对象。

本节将从静态和动态两个角度来对 Snort 的结构和流程进行剖析,揭示它的主要检测机制、模块功能和编程方法。

7.4.1 Snort 的模块结构

首先我们通过图示的方法来介绍 Snort 模块的组成以及相互关系,如图 7-1 所示。从功能上讲,对图中各个模块分析说明如下:

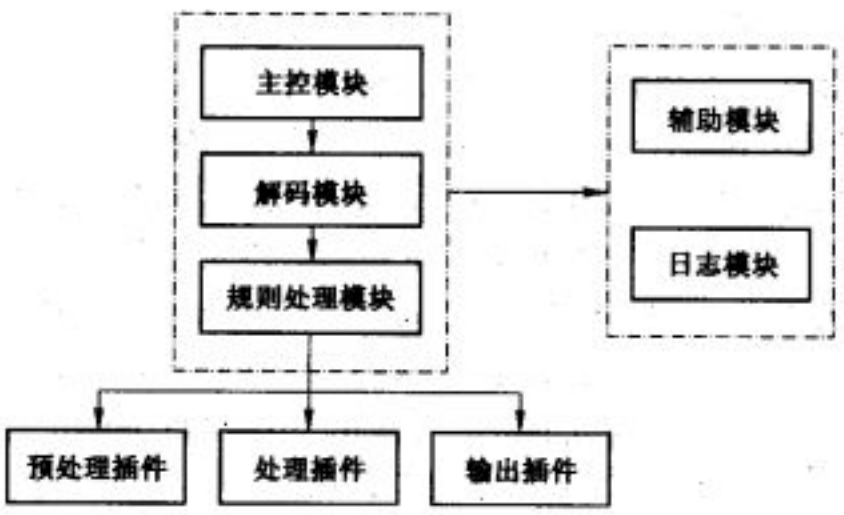

图 7-1 Snort 总体模块图

- 主控模块实现的功能包括所有模块的初始化、命令和解释、配置文件解释、数据包捕获库 libpcap 的初始化,然后调用 libpcap 开始捕获数据包,并进行解码检测入侵。此外,对所有插件的管理功能也属于主控模块的范围。

- 解码模块把从网络上获得的原始数据包，从下向上沿各个协议栈进行解码并填充相应的数据结构，以便规则处理模块处理。
- 规则处理模块实现对这些报文进行基于规则和模式匹配的工作，检则出入侵行为。在初始化阶段它还负责完成规则文件的解释和规则语法树的构建工作。规则处理模块在执行检测工作时使用了三种形式的插件，分别为预处理插件模块、处理插件模块和输出插件模块，它们在文件数量上占有绝对的优势，一共有 47 个源文件(不包括头文件)。主控模块中的插件管理功能实现对所有插件的管理，包括所有插件的初始化和启动等。
- 预处理插件在模式匹配之前进行，对报文进行分片重组、流重组和异常检查等预处理操作。
- 处理插件主要检查数据包的各个方面，包括数据包的大小、协议类型、IP/ICMP/TCP 的选项等，辅助规则匹配完成检测功能。
- 输出插件实现在检测到攻击后执行各种输出和反应的功能。
- 日志模块实现各种报文日志功能，也就是把各种类型的报文记录到各种类型的日志中。
- 在系统运行过程中，还使用了一些辅助模块：树结构定义子模块定义了几种 Snort 使用到的二叉树结构和相关的处理函数，tag 处理子模块完成了和 tag 相关的功能，另外一些子模块也提供了一些公用的函数，如字符串处理等。

7.4.2　插件机制

在 Snort 中运用了插件机制。对于 Snort 来说，插件机制具有以下优点：

- 通过增加插件，使程序具有很强的可扩展性。
- 插件机制简化了 Snort 的编码工作。
- 插件机制使代码功能内聚，模块性强，程序相对易读。

主要的插件模块包括预处理插件、处理插件和输出插件三种，它们通常对应规则中的一个或几个关键字，规则匹配中遇到这些关键字时就会激活相应的插件，以完成相应的功能。下面首先对这三种类型的插件做一简单介绍，然后举例对插件的内部结构进行分析说明，最后对三种插件进行简单的比较。

1. 预处理插件

预处理插件的源文件名都是以 spp __开头的，在规则匹配之前运行、完成的功能主要分以下几类：

- 模拟 TCP/IP 堆栈功能的插件：如 IP 碎片重组、TCP 流重组插件。
- 各种解码插件：HTTP 解码插件、Unicode 解码插件、RPC 解码插件、Telnet 协商插件等。
- 规则匹配无法进行攻击检测时所用的检测插件：端口扫描插件、spade 异常入侵检测插件、bo 检测插件、arp 欺骗检测插件等。

2. 处理插件

处理插件的源文件名都以 sp __开头，在规则匹配阶段的 ParseRuleOptions 中被调用，辅助完成基于规则的匹配检测过程。每个规则处理函数通常对应规则选项中的一个关键字，实现对这个关键字的解释或辅助解释。这些插件的主要功能是：

- 协议各字段的检查，如 TcpFlag、IcmpType、IcmpCode、Ttl、IpId、TcpAck、TcpSeq、Dsize、

IPOption、Rpc、IcmpID、IcmpSeq、IpTos、FragBits、TcpWin、IpProto 和 IpSame 等。

- 一些辅助功能，如 Respond、Priority、PatternMatch、Session、React、Reference 等，这些插件分别完成响应(关闭连接)、严重级别、模式匹配(内容)、会话记录、攻击响应(高级的响应机制)、攻击参考信息等功能。

3. 输出插件

输出插件的源文件名都以 spo _ 开头，这些插件按日志和警告两种类型放入两个列表中，在规则匹配过程中和匹配结束之后调用，以便记录日志和警告。和其他插件相似，它们也对应一个相应的关键字，规则中相应的关键字将激活这些插件的内部结构。

每个插件都有一个安装函数，名称为 SetupXXX( )，如 Spp _ http _ decode.c: SetupHttpDecode( )。这个函数需要放在 Plugbase.c: InitPreprocessors( )中，而且一定要被调用。这些安装函数会在一个列表中注册自己对应的关键字和相应的初始化函数，Http 解码预处理插件安装了两个关键字“http _ decode”和“http _ decode _ ignore”，其相对应的初始化函数是 HttpDecodeInit( )和 HttpDecodeInitIgnore( )，把它们注册到列表 PreprocessKeywords 中。

在解释规则文件时，如果检测到相应的关键字，系统就会使用规则文件中这些关键字后的字符串作为参数来调用相应的初始化函数。初始化函数会根据这些参数初始化插件中状态，并把相应的处理函数注册到处理列表中。这里，当初始化函数 HttpDecodeInit( )完成后，会把处理函数 PreproUrlDecode( )注册到列表中。如果规则文件中没有相应的关键字，就不会触发相应的初始化函数，入侵检测过程中也就不会调用相应的处理函数，也就是不使用这个插件。

在检测过程中，一旦规则匹配成功，就会触发处理函数的执行，预处理器就会在 preprocess 预处理过程中对数据报调用 PreprocUrlDecode 进行处理。

从上面的分析可以看出，输出插件和预处理插件除了注册到不同的列表中之外，其他的过程都很相似；处理插件的过程其实也是大同小异，只是在初始化过程中有所不同而已。输出和预处理插件的初始化通常只有一次，在内存中只有一个实例，所以被注册到一个列表中；处理插件则完成每个匹配规则的一部分功能，所以处理插件为每个匹配规则初始化一次，然后插入到规则树中该规则的列表中。

综上所述，一个插件通常由三个框架函数组成：插件安装函数、插件初始化函数和插件处理函数。插件安装函数　函数名为 SetupXXX( )，一般在程序初始化时调用，是注册插件的初始化函数。插件初始化函数的函数名为 XXXInit( )，一般在解释规则文件时调用，主要功能为完成本插件的初始化，并注册处理函数。插件处理函数的函数名为 XXXXX( )，一般在检测流程中调用，主要在检测过程中完成插件的功能。

7.4.3 libpcap 应用的流程

由于 Snort 定位在轻量级的入侵检测工具上，因此它的体系结构简单，基本上属于事件驱动类型。同时，它又是一个标准的基于 libpcap 库的应用，使用 libpcap 库来捕获网络上的报文，并触发 Snort 的检测过程进行检测。图 7-2 就是一个标准的基于 libpcap 库的应用程序流程。

图 7-2 看上去似乎与 Snort 没什么直接关系，但其实它是 Snort 运行的基础，通过它可以增强对 Snort 底层运行情况的理解。

7.4.4　Snort 的总体流程

如前所述，Snort 有三种运行模式。这里，我们只考虑它作为入侵检测系统运行时的情况，其总体流程如图 7-3 所示。

图 7-2　基于 libpcap 应用的流程

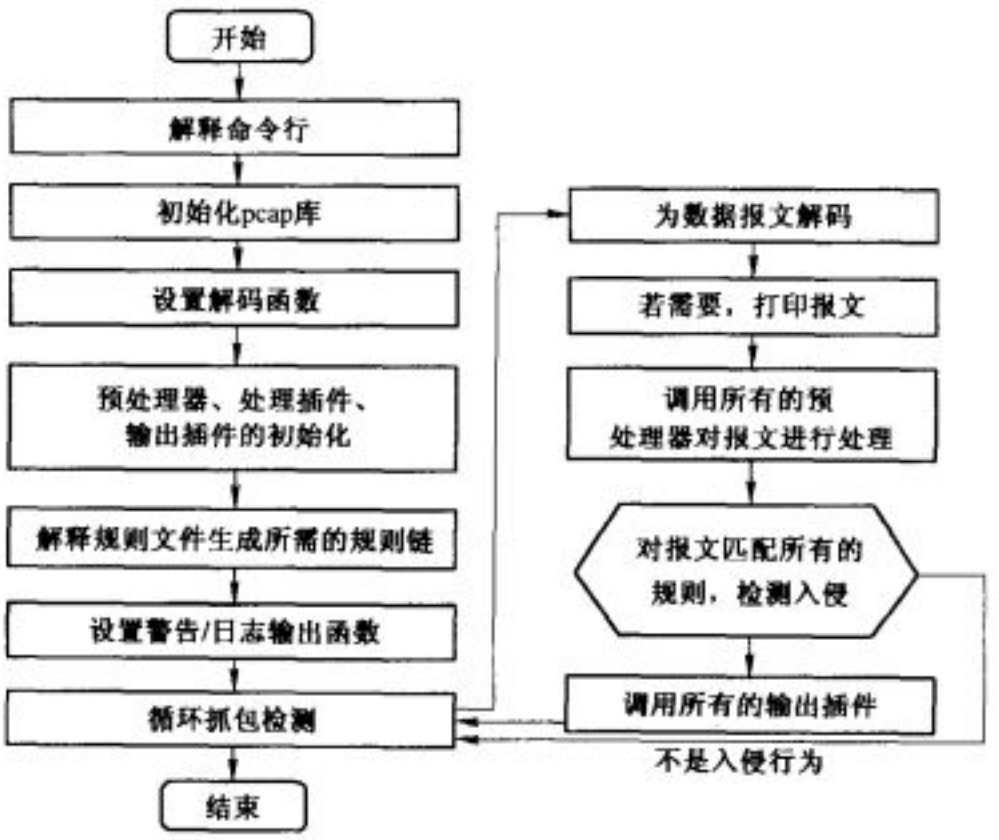

图 7-3　Snort 的总体流程

7.4.5　入侵检测流程

基于规则的模式匹配是 Snort 的核心检测机制。Snort 的入侵检测流程分为两大步：第一步是规则的解析流程，包括从规则文件中读取规则和在内存中组织规则；第二步是使用这些规则进行匹配的入侵流程。下面将依次介绍入侵检测的流程。

1. 规则解析流程

Snort 的规则解析流程很简单，它首先读取规则文件，接着依次读取每一条规则，然后对其进行解析，并用相应的规则语法表示，在内存中对规则进行组织，建立规则语法树。图 7-4 描述了 Snort 的规则在内存中的逻辑关系。

从图 7-4 中可以看出，所有的规则按照规则头排成主链，然后根据规则选项插入到这个链中，构成一棵规则树，这样每一个选项节点对应一条规则。规则头节点主要记录了规则头信息，包括源 IP 端口、目标 IP 端口，并有指针指向下一个规则头节点、附属于它的规则选项列表和规则头列表结构。规则选项节点存放所有的规则选项的信息和处理插件的处理函数列表（opt __ func），分别指向规则头节点的指针和关联选项节点的指针。

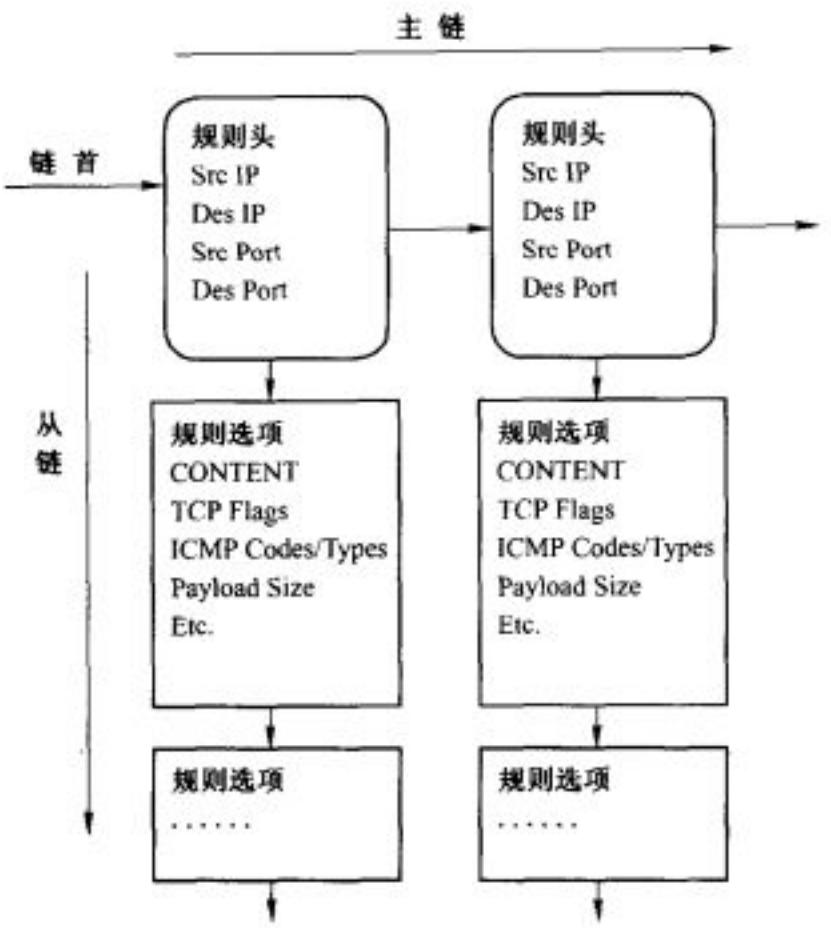

图 7-4 Snort 规则内存表示逻辑图

2. 规则匹配流程

规则匹配的过程就是对从网络上捕获的每一条数据报文和上面描述的规则树进行匹配的过程。如果发现存在一条规则匹配这个报文，就表示检测到一个攻击，然后按照规则指定的行为进行处理(如发送警告等)。如果搜索完所有的规则都没有找到匹配的规则，就表示报文是正常报文。

所有的规则被组织成规则树，然后分类存放在规则类列表中。总体的检测过程归根结底到对规则树进行匹配扫描，并找到报文所对应的规则。对规则树的匹配过程则是先根据报文的 IP 地址和端口号，在规则头链表中找到相对应的规则头，找到后再接着匹配此规则头附带的规则选项链表。

7.5 练习题

一、选择题

1. Snort 是一个跨平台、轻量级的(　　)入侵检测软件。

A. 主机　　B. 网络

C. 综合　　D. 实时

2. 构成 Snort 的三个重要的子系统是(　　)。

A. 数据包解码器　　B. 数据分析模块

C. 检测引擎　　D. 日志与报警系统

3. 在使用 Snort 之前，需要根据网络环境和安全策略对 Snort 进行配置，配置内容主要包

括有(　　)。

A. 设置网络变量　　B. 配置所使用的规则集

C. 配置预处理器　　D. 配置输出插件

4. Snort 可以有的运行方式包括(　　)。

A. 嗅探器　　B. 抓包器

C. 主机入侵检测系统　　D. 网络入侵检测系统

5. Snort 的规则在逻辑上分为(　　)部分。

A. 2　　B. 3

C. 4　　D. 5

6.(　　)属于黑客经常利用的漏洞。

A. 拒绝服务攻击漏洞　　B. 缓冲区溢出

C. 远程命令执行漏洞　　D. 以上全部

7. 入侵检测的规则是指(　　)。

A. 确定 IDS 应用程序如何工作的具体条目

B. 一个完整的特征数据库

C. 一个在安全策略中常规应用的术语

D. 一个分离的应用程序

8.(　　)不存在于 IDS 的应用规则中。

A. 源 IP 地址　　B. 目的 IP 地址

C. 日志条目　　D. 目的端口

二、简答题

1. 简述 Snort 的主要特点。

2. 简述在 Snort 中,使用插件机制的优点。

3. 简述在 Snort 中,处理插件的主要功能。

4. 简述在 Snort 中,预处理器的作用。

5. 简述在 Snort 中,规则解析流程的工作过程。

第 8 章　入侵检测的发展趋势

本章导读：

入侵检测技术是网络安全领域内一门正在发展中的新技术，本章主要是对入侵检测的发展趋势做一展望。

在入侵检测迅速发展的这些年中，我们可以看出安全战略家对主要的技术发展准备不足。例如，十年前，Microsoft 被认为是一个仅限于操作系统的销售商，WWW 技术被认为是出自于欧洲粒子物理研究所(CERN)的一个有趣的思想，仅此而已。在安全领域很少有人认真地考虑过这些技术的安全效果，因为这要付出很大的努力。但是在如今，研究这些技术的安全问题已成为信息时代的主流。

在本章中，我们要分析不同的技术趋势和安全趋势，并且讨论它们对将来的入侵检测功能的影响。

8.1　入侵检测技术现状分析

前面章节中详细介绍了入侵检测技术的基本概念、发展历程、技术方案以及入侵检测系统评估标准、设计原理，并对典型入侵检测系统进行了分析。通过这些介绍和分析，我们可以得出以下结论：

- 入侵检测技术是网络安全解决方案的一个重要组成部分。它不同于传统的安全产品，是从另外一个角度来解决安全问题。研究入侵检测系统具有极其重要的意义。
- 对于入侵检测的研究已经取得了相当的进展。无论是在入侵检测系统的体系结构上还是在检测方法上都在向多样化发展。系统结构上，从原来的单机发展为分布式系统；在检测方法上，从最初简单的匹配到现在的多种分析方法并存。
- 尽管入侵检测技术已经取得了一些进展，但是还远不成熟。现在的商业产品还停留在初级阶段；而研究项目尽管已经取得了一些进展，但是还是处于不断的尝试之中，还远没有达到转化为成熟产品的地步。

通过以上分析可以看出，入侵检测技术的发展仍然任重而道远，需要广大研究人员和工程技术人员不断努力。

8.2　目前的技术趋势

8.2.1　大规模网络的问题

最初的 IDS 都是统一集中式的，运行在网络的单个节点处。这对于小型网络，如只有单一子网的网络是足够了。对于包含了若干子网的大型网络，解决办法就是将 NIDS 部署在网

络的出口处，典型的位置是防火墙背后。但是这样 IDS 系统的流量负载就很大，很难做出实时的复杂检测，而且无法检测内部网段之间的入侵。随着现代网络规模的扩大，网络的拓扑结构也日益复杂，往往包含了多个子网，网络设备包括了路由器、交换机、网桥等，内部网的出口也不只一个。这样集中式的 IDS 就不能适应这种状况。所以越来越多的系统采用了分布式的结构。

8.2.2　网络结构的变化

在网络互联的过程中我们发现交换拓扑结构已经在很大程度上取代了广播拓扑结构。这个变化将会给网络入侵检测带来新的问题。而且，网络技术的发展将会持续影响入侵检测系统所驻留的环境。

随着光纤网络技术和通用网产品的出现，网络带宽以令人难以置信的速度增加（从 1989 年到 1999 年增长超过了 100 万倍），网络访问提供商和电话服务提供商之间的区别将会越来越模糊。

通用网的概念是指所有现有的网络（语音网、Internet、有线电视网和视频广播网）都能够被集成为一个包交换的性能和管理能力显著提高的光纤网络。这个过程将使用波分复用网络交换技术，交换速度可达到千兆位，这种网络流速将会促使访问开销大大降低。

在有些国家，Internet 服务提供商免费向公众提供访问，它们只是从政府部门或企业那里获得一些津贴。随着低价格访问或免费访问的出现，传统的使用调制解调器零星地连接到网络上的拨号联网模型正在转换为持续的访问模型。当网络访问是持续的时候，受到攻击的机会也就会大大增加。这种环境给安全带来了更明显的需求。

8.2.3　网络复杂化的思考

大型网络的另一个特征是网络功能的复杂化。网络内部的流量包括了各种业务，如各种服务 FTP、HTTP、Telnet 等。而对各种服务的入侵手段是不相同的。这样就需要在检测端加载大量的预处理模块和检测模块，极大地加重了检测单元的负担。

8.2.4　高速网络的挑战

IDS 面临的一个重大挑战是高速网络。现在的网络传输速率已经达到了 Gbit/s。而 IDS 的基础是获得完整的网络数据流，否则是无法完成正常检测的。而现在的 IDS 的软件抓包器速率一般只能达到 Mbit/s，远远不能满足应用。这个问题的解决是当前网络安全界研究的重要课题之一。现有的解决方法有采用硬件采集器等。

8.2.5　无线网络的进步

家庭网络访问的普及提出了对瘦客户端集中数据管理服务的需求。这些服务包括集中管理的应用程序和无缝访问。在集中管理的应用程序的情况下，访问的价格包括使用不同软件应用程序的能力，所有这些都由访问提供者来管理。在无缝访问情况下，用户可以像使用桌上电脑一样容易地使用蜂窝电话或手持个人数字设备。

高速无线网络的增长将驱动无缝访问。在无线网络领域中共有 4 个主要的竞争者提供从

1.6MB/s～46MB/s的数据传输率的服务。这些无线网络将会随着不同设备控制协议的出现而成倍增长,从而使智能家庭能够通过使用远端控制器和通信设备以及处理可视电话信息的集成个人数字设备来管理金融账号并进行金融交易。因此,无线网络的增长导致了分布式计算的出现。

8.2.6 分布式计算

第1代计算定义为多个用户使用一台计算机。第2代计算定义为一台计算机一个用户。第3代计算是分布式计算,提倡一个用户使用多台计算机并且可能改变现在流行的大多数计算模式。分布式计算的概念是指用来计算的计算机可以放置在任何地方,但是在物理的环境中必须是透明的和可用的。而且,这些计算机通过无线网络互联,能够协调它们的活动,能够交换信息,能够适应环境的变化。

多年来,人们一直认为物理环境是安全的,但突然发现他们很容易被对手通过网络连接进行攻击,这些技术对安全的影响是深刻的。就像电话销售商和恶作剧的打电话者已经改变了人们对电话服务的态度一样,分布式计算的滥用和任意的网络访问可能会在很大程度上改变人们对网络服务的态度。

8.2.7 入侵复杂化

从某种程度上来讲,入侵技术一直领先于安全技术的发展,两者相互推动、互相促进。随着网络的日益普及以及各种黑客工具的蔓延,入侵的复杂化趋势也越来越明显,向着网络化、分布式、隐蔽化方向发展。入侵检测技术必然要适应这种趋势。从集中研究基于主机的入侵检测系统到基于网络的入侵检测系统的出现,就体现了这样一种要求,单纯依靠一个节点的日志信息已经无法判定入侵,需要从更底层来进行检测。入侵的另一个发展趋势是分布式入侵,这就要求综合多个节点的信息来进行判定。

8.2.8 多种分析方法并存的局面

对于入侵检测系统,分析方法是系统的核心。目前针对入侵检测系统有很多种入侵检测分析方法,但各种分析方法各有利弊,大多数分析方法只适合某些种类的入侵,没有一种方法能够解决所有的问题。针对这样的情况,作者认为,在目前的情况下,多种分析方法综合运用才是可行之道。那么如何将各种分析方法有机结合起来,构建出高性能的入侵检测系统呢?这也是一个值得研究的问题。

8.3 未来的安全趋势

每一个技术进步以及和这个技术进步相关的增值(如Internet增长带来的增值)都产生了相应的安全问题。现代系统的复杂性给人们设计处理这种环境的改进性方法带来了挑战。

8.3.1 管理

在计算机安全的早期,持不同观点的专家们对计算机安全的实质进行了激烈的争论。有

些专家认为计算机安全纯粹是一个技术问题，它可以通过软件工程来实现，可以通过确定是否达到了一定的保险级别对软件进行评估来解决。

但是，另外一些专家认为计算机安全是一个纯粹的系统管理问题，因为不论系统的技术质量如何，用户和管理员将一直是大多数计算机安全问题的主要来源。随后的软件工业和相关的计算机安全问题的进步证明了两种观点都不完全正确，也不完全错误。

如果与负责保护大型组织系统的安全专家讨论安全问题，就会发现他们关于组织内的安全措施的第一个抱怨就是它们太不容易管理了。易管理性是指对技术和产品的使用应达到两个目的：增强系统的易用性，同时减少费用。

为了使将来的安全产品便于管理，特别是当这些安全产品保护的环境不便于管理时，需要花费多大的代价呢？专家们提供了几个原则，他们认为为了使安全性能和管理平台很好地结合，应该考虑下面的问题：

- 可管理的系统是简单的。应该使用一定的技术手段和良好的技术实现来消除尽可能多的复杂性。
- 可管理的系统是能够远程访问的。这个要求带来了对安全产品的需求，但是也确实从安全产品中获益。当分析家对系统事件有类似的广泛了解之后，许多种攻击策略就都是很明显的了。
- 可管理的系统应该被设计为易于使用的。这个要求意味着即使当系统很复杂时，系统信息的表示也能够使得用户以方便的形式来访问系统。
- 可管理的系统是可靠的，并且即使系统发生了故障，也能够很容易地解决问题。
- 可管理的系统都假定系统是可变的并且提供了一些变化的方式。

除了这些原则，下面的事实对于复杂系统的安全管理也是非常重要的：

- 在可能的地方，安全应该对用户透明。这个实现主要用来减少与用户不会使用安全机制或使用安全机制时发生错误的相关问题。
- 在安全不透明的地方，安全机制操作起来应该非常简单。操作简单可能意味着基于标识或生物学特性。而不意味着与机器相关的每 10 天换一次很多个字符长的口令。

8.3.2　保护隐私安全

Internet 的互联性带来了与个人隐私权相关的问题，如匿名性、隐私性和安全性。

匿名性是指“不能够被命名或标识的性质或状态”。隐私性定义为“免于非授权入侵”。安全定义为“免于危险”。Internet 上的一些组织认为匿名性和隐私性是一样的。也有一些人认为尽管安全性对于保证隐私性来说是必要的，但它同时也带来了对隐私性和匿名性的威胁。

事实上，增强隐私性并不一定就需要匿名性。也许隐私代表的是对匿名性的控制，而不是严格地保证匿名性。毕竟，为了增强安全性，匿名性直接取消了一些机制中对防止非授权访问进行保护所必要的可说明性的概念。另一方面，安全性应该被设计用来保护隐私，也就是说，安全性的一个主要目标就是保护用户的隐私性，只有在事先允许的情况下才能泄漏用户的身份。

这种隐私保护过程可以通过一些机制来对用户进行授权但不进行认证。例如，在金融交易中，拥有现金确实可以证明用户拥有货物或服务而不必将用户的身份泄漏给与用户作交易

的商人。电子商务非常需要支持没有认证的授权的金融交易。

入侵检测的一些方法使得安全性和隐私性的平衡很容易达到。例如，集中于交易的过程（也就是不会泄漏通信内容的传输模式和其他信息），而不是集中于交易的内容（真正要传输的消息），就能够使得在不牺牲用户隐私性的前提下，确定什么时候发生了什么问题。

8.3.3 加密

加密是古典信息安全的基础，同时也在虚拟专用网、硬件连接加密器、加速卡和介质加密软件中变得越来越常用。这种技术将改变网络入侵检测的外观，从而使基于内容的网络数据包分析器和其他基于流的信息源失效。

尽管加密的广泛使用可以解决网络安全中的一些重要问题，但是针对加密软件和硬件的攻击工具可以开发出来。这些攻击将从交易的端点收集信息，特别是关于密钥的信息，在这些端点处数据是不加密的。检测这些攻击所需要的信息与入侵检测系统当前使用的信息是不同的。用密码学操作对端点进行保护可能会成为入侵检测系统的一个重要功能。

8.3.4 可靠传输信任管理

最后，Internet 在商业上的使用强调了安全机制作为可靠传输代理的角色，而没有强调作为信任管理机制的角色。这就增强了对强识别认证匿名保护授权机制、对审计追踪信息和其他交易日志信息的保护的需求。这也影响了检测的基本内容，因为商业规则将规定必须要检测和阻塞的活动类别。这种趋势暗示了商业保险行业要在一定程度上加入到制定安全标准的过程中来，至少要加入到在 Internet 上和其他系统环境中进行的商业活动的标准制定过程中来。

8.4 入侵检测的前景

在构成将来的系统安全的大环境中，我们来看一看入侵检测及其支持的功能适用于什么地方。贯穿本书，可以看到无论是现在还是将来入侵检测（和与其相关的监测功能）所扮演的持久的角色。这些角色可以监测目标系统中其他安全机制的功效。让我们进一步讨论一下入侵检测的前景。

8.4.1 入侵检测的能力

在最好的环境下，入侵检测系统需要做什么？对于这个问题，网络安全专家们有很多不同的回答，这些回答有些集中在系统的分析能力上，而另外一些则侧重于检测系统本身的管理和协调操作。

优化的入侵检测系统应该能够进行基于事件语义的检测，而不是基于事件语法的检测。这种方法弥补了当前在安全政策和检测政策之间的差距。例如，给定一个语义检测引擎，告诉入侵检测系统去“检测所有的违反访问控制的访问”，那么入侵检测系统就会去做，而不管被检测系统使用的是什么平台、什么协议或什么数据类型。这种类型的操作是对当前检测状态的一个极大改进，在当前的检测中，检测目标需要复杂的、特定的与操作系统相关的检测特征。

这种能力的另外一个优点就是在检测“多态”或“变异”（就像多态病毒那样）的攻击时，检

测器不仅能够识别出变异后的攻击，而且也能识别出原始的攻击，这样就建立了一个连接这两种攻击的用来提供证据的追踪信息。

另外一个优化的特点是入侵检测功能与通用网络管理结合起来。这种紧密的结合使入侵检测系统能够进行良好的趋势分析，可能也会及时预测有威胁的攻击，从而阻塞或防止这些攻击。从与法律执行官和事件处理专家的讨论可以看出，将来的入侵检测系统必须要包含支持入侵调查的特点。

由于全世界各地的法律处理过程存在细微差别，因此检测功能必须能够收集可靠的信息。这些信息必须得保证在从开始产生到作为法庭证据而使用的过程中不能被修改和破坏，被召集到一起的陪审员和调查员（非技术人员）要在这些信息的基础上作出决定，检测系统的功能应该能被他们所理解。

改进入侵检测系统的易用性和易管理性，特别是在多设备（超过 1 万个主机）的情况下，通常被认为是未来入侵检测系统所必需的。这样的新系统在被设计时，必须得知道哪些入侵检测功能是完全自动的，哪些是需要人工干预的。新系统也包括一个丰富的界面选项集，允许用户以不同的外观来检查事件模式，从而选择感兴趣的条目。

8.4.2　高度的分布式结构

将来的趋势是入侵检测高度分布式监控结构的使用。这种方法将使用许多具有不同定位策略的自主代理。例如，根据使用的策略，每一个代理可以定位一个特定的事件类型、特征类型、平台或过程，可以有不同的策略来管理分析功能并反映功能的存放。最可能的是，这样的代理可以和其他信息源（操作系统和基础设备中的日志机制）同时存在，并且也可以被一个监督进程所管理。这个监督进程将把从代理得到的数据与从其他数据传感器得到的数据关联起来，从而识别出细微的问题所在。

分布式监督和分析体系结构也能很好地适应实现某些免疫系统的入侵检测方法。例如，许多代理都会检查系统，寻找接触关键文件的反常过程。每一个代理都按照一个特定的攻击特征来衡量进程的活动。如果一个代理发现了具有某种攻击特征的进程，它就会修改这个进程，也可能阻止这个进程的处理速度。由于这个进程会激发许多代理，每一个代理都会使这个进程的处理速度慢一点，所以这个进程的处理速度会慢到足以被人或被某种代理记录下来，代理记录关于这个进程的信息并将其杀死。

8.4.3　广泛的信息源

随着当前网络带宽和节点的迅速增长，入侵检测系统必须监控的原始资料也不断增长。这种情况使得有必要去收集和分析入侵检测系统的组件。原始资料可能超出了最经常使用的主机信息和网络信息的范围。例如，信息可以从硬件设备、瘦客户端、通信连接以及安全体系结构中的其他部分中获得并得到加强。而且，支持商业过程和法律过程的关键是从网络结构中收集信息，进行冗余监控，使用分析机制来标识出对检测到的攻击的确证信息。

8.4.4　硬件防护

可能发生的另一个趋势就是硬件版本的入侵检测系统和安全网络工具箱集成在一起。这

种设备将定位于家庭市场和小型企业市场，以使客户能够处理与持续连接到 Internet 上相关的安全问题。集成的安全和网络工具箱可能会包括网络接口硬件（保护集线器和路由器）、防火墙、连接加密器、Web 服务器和其他用来加强更快更安全连接的功能。

8.4.5 高效的安全服务

就像电话系统一样，中央通信和网络提供商将需要包含入侵检测。这些变化将随着网络服务应用的出现而出现。

当家庭网络服务作为集成网络访问服务包的一部分被提供时，入侵检测可以以多种姿态存在。首先，如果入侵检测作为标准的网络管理的一部分在网络层实施时，它应该对终端用户是完全透明的。

第二，如果入侵检测是在家庭的水平上提供给用户的，那么它可能是作为集成网络访问包的一部分提供给用户的可选的集中化服务，就像今天电话服务中的呼叫等待服务或呼叫者 ID 服务。

最后，如果网络访问应用程序市场处于不规范状态（具有自由竞争的特点），购买决定将依赖于应用程序在合适的价格下提供高质量的服务的能力。

在这种情况下，应用程序使用入侵检测来确保高质量的服务。在任何情况下，入侵检测都将和网络管理工具结合得越来越紧密，并且允许网络管理员根据局部或全局需求来设置，监视和检测内容的特点将越来越鲜明。

8.5 练习题

一、选择题

1. 使用(　　)方案来应用修补措施，从而停止别人对已知弱点的利用。

A. 防火墙　　B. 加密

C. 确认　　D. 对于弱点进行修补

2. (　　)不是动态安全模型。

A. Policy　　B. Protection

C. Detection　　D. Request

3. 计算机安全技术包括(　　)。

A. 保密性　　B. 完整性

C. 可控性　　D. 以上全部

二、简答题

1. 简述入侵检测技术的现状。
2. 简述目前的技术趋势主要体现在哪些方面？
3. 简述网络的大规模化和复杂化给入侵检测带来的问题。
4. 简述未来的安全趋势。
5. 简述入侵检测的前景。

参考文献

1. 蒋建春,冯登国．网络入侵检测原理与技术．北京:国防工业出版社,2001

2. Rebecca Gurley Bace 著,陈明奇等译．入侵检测．北京:人民邮电出版社,2001

3. 韩东海,王超,李群．入侵检测系统实例剖析．北京:清华大学出版社,2002

4. 戴英侠,连一峰等．系统安全与入侵检测．北京:清华大学出版社,2002

5. 周学广,刘艺．信息安全学．北京:机械工业出版社,2003

6. 李春玲．基于 Agent 的分布式入侵检测[硕士论文]．北京:北京理工大学,2003

7. 翟洪波．入侵检测系统研究[硕士论文]．北京:北京理工大学,2003

8. 岳治宇．可扩展的入侵检测框架设计与实现[硕士论文]．北京:北京理工大学,2002

9. 启明星辰．入侵检测技术培训教材．北京:启明星辰信息技术有限公司培训认证部

10. 国家信息化安全教育认证管理中心．信息安全知识读本．北京:长安出版社,2003

11. 启明星辰．天阗网络入侵检测系统白皮书．http://www.venustech.com.cn/pands/ tiantian __ index.htm,2003

12. 中科网威．"天眼"网络入侵侦测系统技术白皮书．http://www.netpower.com.cn/index.asp,2003

13. 绿盟．冰之眼网络入侵检测系统．http://www.nsfocus.com/homepage/products/nids.htm,2003

14. 蒋建春,马恒太,任党恩,卿斯汉．网络安全入侵检测:研究综述．北京:软件学报,2000

15. http://www.snort.org/

16. http://www.iss.net

附录　选择题答案

第 2 章

1. A　2. C　3. B　4. C　5. B　6. D　7. D　8. D　9. C　10. A

第 3 章

1. C　2. A　3. B　4. C　5. B　6. A　7. C　8. D　9. B　10. D

第 4 章

1. AC　2. ABC　3. B　4. A　5. D　6. D　7. A　8. D　9. D

第 5 章

1. A　2. ABC　3. C 4. BD　5. B　6. D　7. C　8. D　9. C

第 6 章

1. B　2. C　3. A　4. AB　5. B　6. D

第 7 章

1. B　2. ACD　3. ABCD　4. ABD　5. A　6. D　7. A　8. C

第 8 章

1. D　2. D　3. D